ISSN 0869-3889

МОНИТОР

Версия 1. 92 (3)

журнал для программистов—практиков

Страсти по WINDOWS

CLIPPER 5.0 – новые возможности

Технология создания оконного интерфейса

Бутовые вирусы на марше

Этюды для программистов

Выходите на связь

Все о TURBO C и C++!

"Библиотека журнала "Монитор"
представляет первую книгу

"Функции Turbo C, C++".

Более чем на 300 страницах тщательно подобрана информация о библиотеках C, Turbo C, C++.

Вы получите:

- детальное **описание более 400 функций;**
- примеры, поясняющие работу **каждой функции;**
- список совпадающих функций Turbo C, C++, Microsoft C;
- демонстрационные программы, поясняющие работу групп функций;
- указатели для быстрого поиска по имени функции или ее назначению;
- введение в объектно-ориентированные средства Turbo C++;
- современный дизайн.

Эта книга может служить справочником и учебником. Мы позаботились о том, чтобы Вам было удобно и приятно ее читать.

Если вы программируете на Turbo C и C++, интересуетесь вопросами совместимости с Microsoft C, то эта книга для Вас.

Вышлите 3 экз. кн
„Функции Турбо С

ПОЧТОВАЯ КАРТОЧКА

Куда 117419, г. Москв
а/я 765

Кому „Монитор"

117419

Пишите индекс предприятия связи места

Спешите! Тираж ограничен.

Ориентировочная цена книги 24 рубля
Оптовым покупателям предоставляется скидка.
Заказы направляйте по адресу: 117419, Москва,

МОНИТОР

Выпуская в свет первый номер журнала "Монитор" за 1992 год, мы считаем необходимым сказать нашим читателям следующее.

Первые два номера журнала вышли с опозданием. В связи с этим третьего номера за 1991 год не будет, зато в 1992 году вместо планировавшихся восьми выйдут девять номеров "Монитора". Читатели, оформившие подписку на три номера журнала, получат два номера за 1991 год и первый номер за 1992 год; те, кто подписался на шесть номеров — два номера за 1991 год и первые четыре номера за 1992 год. Просим Вас учитывать это при продлении подписки. Подписной купон Вы найдете на стр. 61.

Мы всегда рады звонкам и приходу наших читателей. Мы приглашаем к сотрудничеству как авторов статей по программированию, так и авторов, которые могут предложить серьезные экономические обзоры положения на рынке компьютеров и программного обеспечения, статьи по правовым вопросам, свои размышления о месте и роли компьютеров в жизни современного общества.

Мы будем очень благодарны тем читателям, которые не пожалеют времени, чтобы заполнить и отправить в адрес редакции опубликованную на стр. 50 анкету. Результаты анкетирования помогут нам сделать журнал более качественным и интересным для Вас.

Мы издаем журнал не только для Москвы, где литература по программированию все же более доступна. Поэтому, если в Вашем городе "Монитор" еще не продается, и Вы хотите заняться его распространением, то мы готовы к сотрудничеству с Вами.

Мы хотим особенно обратить внимание наших читателей на два материала из числа опубликованных в этом номере.

Первый из них - статья В.Водолазского "Бутовые вирусы на марше". Она открывает серию статей, посвященных устройству компьютерных вирусов и методам борьбы с ними. Мы отдаем себе отчет в том, что знания такого рода могут быть использованы как во благо, так и на зло, однако считаем, что польза от того, что читатели нашего журнала овладеют элементарными приемами самообороны от вирусов и станут менее зависимы от авторов антивирусных программ, значительно перевешивает возможный вред. Квалифицированный программист, задавшийся целью разобраться, как устроен вирус, и написать собственный, справится с этой задачей и без наших публикаций, а неквалифицированный новоиспеченный Герострат просто выведет из строя собственный компьютер. Мы знаем, что некоторые специалисты по компьютерным вирусам придерживаются иной точки зрения, и, если они пожелают, им будет предоставлено слово в нашем журнале.

Второй материал, который мы хотели бы выделить, - заметки о прошедших в Москве собрании Ассоциации поставщиков программных продуктов (АППП) и конференции по правовой защите программного обеспечения. Проводившаяся многие годы в нашей стране государственная политика нелегального копирования западных компьютеров привела к подрыву собственной компьютерной промышленности. Точно так же существующая общественная практика терпимого отношения к нелегальному копированию программ может привести к тому, что программистам в России не останется другой работы кроме русификации и сопровождения импортного программного обеспечения. А ведь количество программистов в нашей стране и уровень их квалификации таковы, что вполне позволяют не только насытить внутренний рынок, но и сделать производство программ важной статьей российского экспорта.

Разумные законы по охране программного обеспечения защищают не только интересы фирм - поставщиков программ, но и наши с вами рабочие места, и на их скорейшее принятие возлагаются большие надежды. Однако еще древние римляне говорили: "Quid sunt leges sine moribus?" - "Что значат законы без добрых нравов?" А добрые нравы, то есть отношение к нелегальному копированию программ, как к воровству, могут установиться только в результате наших общих усилий. Тема правовой защиты программного обеспечения также будет постоянно присутствовать на страницах "Монитора".

Журнал для программистов-практиков

Редколлегия:

Д.Э.Бройтман
(главный редактор)

М.И.Гетманская

И.Г.Гун

И.Н.Исаев

С.В.Макельский
(главный художник)

А.В.Орехов
(издательский директор)

А.В.Свириденко
(отдел маркетинга и рекламы)

А.В.Семенников

Л.И.Устинова

А.А.Якубенко
(ответственный секретарь)

В оформлении номера использованы рисунки

К.З.Гусалова

Подписано в печать с оригинал-макета 21.02.92.
Формат 60x90/8.
Печать офсетная. Печ.л.8.0.
Изд. N1971.
Цена договорная

Издатель:
Научно-техническое общество

Софт·Москва

Для писем: 117419, г.Москва, а/я 765, тел. 237-21-36

© "Монитор", 1992

Операционные системы

Д.Тарараев. **Microsoft Windows: новый подход в программировании** 4

Рассматриваются основные принципы построения Microsoft Windows 3.0. Обсуждается взаимодействие Windows с оперативной памятью, операционной системой и программным обеспечением.

П.Емельянов. **Страсти по Windows** 8

Базы данных

А.Чернышков. **Clipper 5.0 - новые возможности Часть I** 9

Первая статья из цикла статей, посвященных системе программирования Clipper 5.0.

Панорама

PC: история и сегодняшний день 16

Технологии программирования

М.Черкашин. **Сравнительное жизнеописание двух чемоданов с инструментами** 17

На основе анализа особенностей языков Турбо Паскаль 5.5 и C++ рассматриваются основные идеи объектно-ориентированного программирования

Лицей

С.Курковский. **Технология создания оконного интерфейса** 22

Рассматриваются вопросы создания текстовых окон на языке Турбо Си. Управление текстовыми окнами ведется с использованием системных прерываний и манипулирования видеопамятью. Разработана программная реализация интерфейса для управления текстовыми окнами, структурированная в виде линейного списка.

Дайджест

Персональные компьютеры становятся товаром широкого потребления, а их производители вынуждены менять стратегию 28

Тезисы статьи из журнала "Economist" (2 – 8.11.1991)

Всякая всячина

В.Водолазкий. **Программа распечатки имен текущих каталогов** 29

Программа на ассемблере, которая выводит на дисплей имена рабочих каталогов для всех дисковых накопителей.

Терроризм

В.Водолазкий. Бутовые вирусы на марше **32**

В статье на примере вируса "6-е Марта" демонстрируется техника борьбы с бутовыми вирусами

Коммуникации

Е.Равич. Выходите на связь. Часть I **36**

Первые шаги в области компьютерной связи для пользователей персональных компьютеров.

Дайджест

Зарубежная компьютерная хроника **40**

Аппаратура

Т.Макогон. Портативные компьютеры **42**

Менее чем за год технология портативных компьютеров совершила качественный скачок к моделям, сочетающим мощность с компактностью.

Этюды для программистов

Д.Бройтман. Этюд 2. Рекурсия или не стреляйте из пушки по воробьям **46**

Несколько классических рекурсивных задач.

Эссе

А.Халамайзер. Homo computerus **51**

Общение с ЭВМ приводит к изменению самих людей, их поступков, привычек, характеров и даже судеб.

Корпорации

А.Ларионов. Про то, как фирма IBM с фирмой Microsoft поссорилась **54**

Рассматриваются взаимоотношения двух титанов компьютерной индустрии – IBM и Microsoft.

Бизнес

Д.Бройтман. Рынок программных продуктов в России: первые шаги **58**

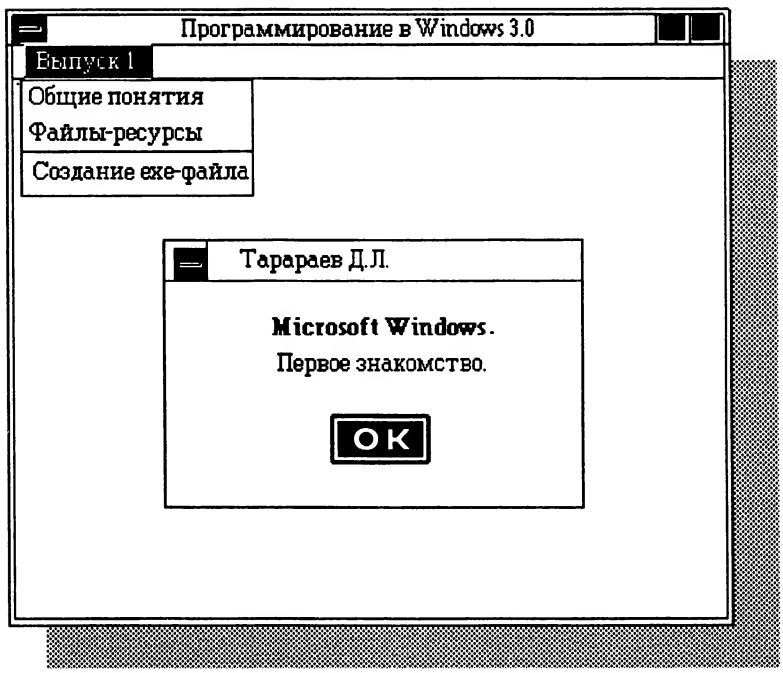
С.Воробьев, С.Лаптев. Итоги торгов Компьютерной секции Московской товарной биржи в ноябре-декабре 1991 года **61**

А.Халамайзер. Перестройка в халифате и в хозяйственном аппарате **63**

А.Свириденко. Merisel в Москве. Время крупных посредников пришло **64**

Microsoft Windows : новый подход в программировании

Д.Тарараев



Благодаря многозадачности и простоте обращения MS Windows стала удобным рабочим столом, на котором всегда находятся под рукой все необходимые инструменты

На протяжении всей эволюции IBM-совместимых персональных компьютеров от процессора 8086 до 80486 основным программным обеспечением, служившим интерфейсом между процессором и программами, был DOS. Параллельно с эволюцией технических средств DOS также развивалась от первых версий с ограниченными возможностями работы с памятью и медленным выполнением функций до DOS 5.0, имеющей широкий набор драйверов работы с расширенной (extended) памятью, таких как RamDrive.Sys, Xma2Ems.Sys и т.д. Но эти изменения не затронули основы операционной системы IBM PC – однозадачности. Тем временем аппаратное обеспечение уходит далеко вперед и DOS не может полностью использовать такие новые возможности, как:

- значительное превышение быстродействия процессора над рабочей частотой периферийных устройств, ПЗУ, винчестера;
- работа процессора в защищенном режиме (protected mode);
- возможность расширения оперативной памяти до 8 Мбайт, а в компьютере с 386 процессором до 16 Мбайт.

Каким же образом использовать эти возможности? Однозадачная операционная система этого сделать не может. Возникает необходимость преодолеть этот барьер, но в то же время сохранить совместимость с огромным количеством созданного до сих пор программного DOS обеспечения. Таким качественным скачком в системном программном обеспечении PC стал выход в свет Microsoft Windows 3.0 в мае 1990 года. MS Windows сразу отвоевала у DOS множество областей при-

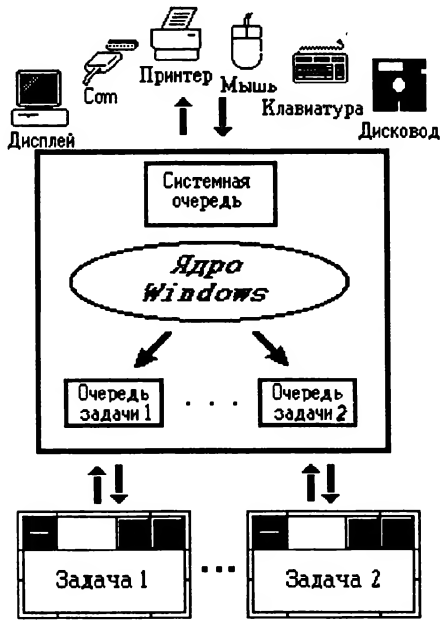


Рис.1. Работа с несколькими задачами в системе Microsoft Windows 3.0

кладного программирования.

В адрес Microsoft Windows было много положительных отзывов. Благодаря многозадачности и простоте обращения она стала удобным (с точки зрения пользователя) "рабочим столом", на котором в виде нескольких задач всегда находятся "под рукой" все необходимые инструменты: текстовый редактор, электронные таблицы, калькулятор, часы и т.д. На 386/486 процессорах можно выиграть время за счет одновременного выполнения задач, требующих длительной обработки, таких как форматирование дискет, архивация файлов, компиляция программ.

Однако кроме удобства для пользователей, Windows пытается решить еще одну серьезную проблему: она позволяет выйти на многозадачность, как на новый уровень программирования.И сейчас мы хотим поговорить о более серьезных перспективах этой, по праву занявшей достойное место в мире программного обеспечения, системы. После первого знакомства с Windows, красивых демонстраций одновременного выполнения нескольких программ, обычно поражающих воображение программиста DOS, возникает некоторая нерешительность перед масштабом Windows программ: с чего же начать знакомство с системой? Предлагаем начать с начала...

Пользователь IBM PC привык к одномерным программам DOS : все задачи выполняются последовательно и результаты работы также последовательно отображаются на экране. Windows же устанавливает качественно новый уровень работы – многомерность, которая выражается, во-первых, в многозадачности, во-вторых, в многооконности (не будем забывать названия системы Windows – Окна). Рассмотрим несколько базовых понятий системы.

Прежде всего в отличие от DOS в Windows может быть запущено одновременно несколько программ, каждая из которых называется процессом или задачей (рис.1). Процессы, выполняемые в данный момент времени, называются активными. У каждой задачи есть свое окно, где происходит отображение ее работы. Но так как устройство ввода одно на несколько задач, то существует понятие текущего (активного) окна, которое и воспринимает прерывания ввода/вывода.

Многомерность также проявляется в общении между процессами. Процесс может быть независимым, то есть выполняться без связи с другими задачами, и зависимым. Общение между задачами может происходить двумя способами.

1. Каждый процесс может использовать общую для всех задач область памяти – ClipBoard (буфер), чтобы размещать данные, которые по необходимости будут использовать другие задачи.

2. Есть также возможность организовывать напрямую связь между процессами – канал, по которому одна задача может управлять другой, обмениваться данными и т.д. Этих каналов может быть несколько у каждого процесса.

Еще одно важное понятие Windows – ресурс. В отличие от DOS, где программа работает непосредственно с реальными устройствами компьютера через функции DOS и BIOS или прямым обращением в порты, Windows-программа взаимодействует с реальным устройством через операционную систему, где это устройство представлено в виде программного ресурса. Понятие системных ресурсов распространяется на все периферийные устройства компьютера: модем, сеть, COM-порт, принтер, звуковое устройство и т.д. Такая организация объясняется прежде всего необходимостью отслеживать одновременные обращения нескольких задач к одному ресурсу.

Схема ресурсного управления придает программе важное свойство – свойство переносимости, т.е. программа сможет работать на машинах с разной конфигурацией, например, с разными видеоадаптерами. В программе вы используете абстрактный ресурс – дисплей. Windows сама связывает его с установленной картой адаптера, и в большинстве случаев вам не надо предусматривать в программе различные типы мониторов. То же относится к принтерам, звуковым устройствам и другим ресурсам.

Позвольте также упомянуть еще некоторые преимущества Windows, представляющие, как нам кажется, большую ценность для программиста.

1. Система задействует все платы расширенной памяти от 1 Мбайта до 8 (16) Мбайт в качестве оперативной памяти.

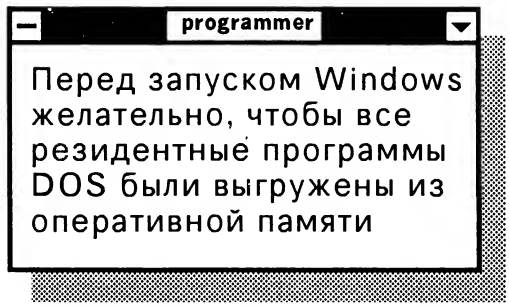
2. Программа Windows не загружается целиком в память. Существуют так называемые процедуры DLL (Dynamic Linking Library), которые загружаются во время выполнения программы и выгружаются из памяти после их использования. Это, во-первых, позволяет работать оперативной памяти в оптимальном режиме , а во-вторых, с точки зрения программиста, через DLL библиотеки можно связывать модули, написанные на различных языках программирования. Например, программа на Borland C++ может вызывать процедуры DLL библиотеки на Microsoft C.

3. Windows использует винчестер для хранения неактивных процессов в 286 процессоре, а при работе на 386,486 процессорах в защищенном режиме на диске хранятся виртуальные страницы памяти (swapping), что делает оперативную память теоретически ограниченной только размером диска.

4. Графический интерфейс вывода, включающий в себя окна, меню, кнопки, курсоры, шрифты и т.д., поддерживается Windows на системном уровне, поэтому работа с ними в программе сводится к использованию соответствующих встроенных процедур Windows. Внешний вид окна в системе Windows изображен на рис.2.

5. Обработка прерываний устройств ввода: мыши, клавиатуры и т.д. происходит в системе, а затем программист имеет к ним доступ в программе в удобной для него форме.

После того как вы ознакомились с основными достоинствами Windows, приведем некоторые пользовательские сведения о системе.



Загрузка MS Windows

Система загружается из DOS при помощи файла Win.com, который загружает ядро Windows – Kernel.exe и программный графический интерфейс (Program Manager). Состав ресурсов системы и параметры загрузки содержатся в текстовых файлах конфигурации – Win.ini и System.ini. Перед запуском желательно, чтобы все резидентные программы DOS были выгружены из оперативной памяти, также для ее лучшего использования не рекомендуется включать в Config.sys драйверы DOS, работающие с расширенной (Extended) и расширяемой (Expanded) видами памяти, такие как VDisk, Xma2Emm и т.д. При установке Windows добавляет в Config.sys два драйвера.

1. HiMem.sys – драйвер Extended памяти, который позволяет Windows пользоваться всеми ресурсами оперативной памяти в одном из трех режимов, речь о которых пойдет в следующем разделе.

2. SmartDrv.sys – драйвер, по работе аналогичный Cash-памяти и учитывающий особенности работы в MS Windows.

Работа системы с оперативной памятью

В Windows существуют три способа организации оперативной памяти.

1. Реальный режим (Real mode). В этом режиме Windows работает на PC XT и AT с оперативной памятью, не превышающей 1 Мбайт. К памяти возможен доступ прямой адресацией.

2. Стандартный режим (Standart mode) – это базовый режим работы Windows 3.0. Он организуется на AT, PS/2 с памятью более 1 Мбайта. При этом операционная система организует так называемую банкуемую память (Banked memory). Оперативная память делится на банки, в каждом из которых действует прямая адресация. В этом режиме возможен запуск 5-6 небольших задач, причем выполняться будет только текущая. Сегменты памяти неактивных процессов могут записываться на винчестер для временного хранения при недостатке оперативной памяти (swapping).

3. Расширенный режим (Enhanced mode). Возможен на AT, PS/2 с процессорами 80386/486 и оперативной памятью не менее 2 Мбайт. В этом случае Windows органи-

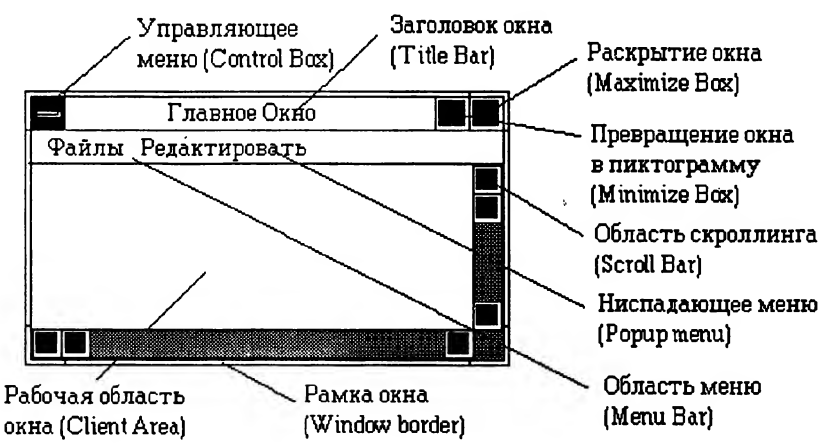


Рис.2. Внешний вид окна в системе Windows

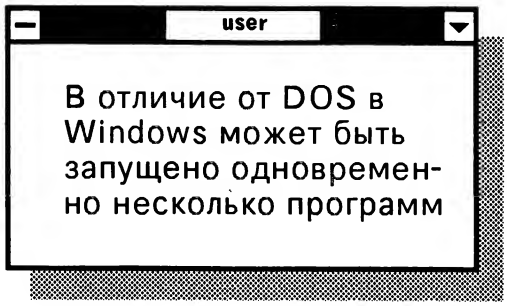
зует постраничную память, причем процессор работает в защищенном режиме (Protected mode). Виртуальные страницы памяти могут храниться на жестком диске, количество процессов – до 10 (затем следует существенная потеря в скорости выполнения задач). Например, при одновременной работе пяти процессов на AT 386 с 4 Мбайтами их производительность меньше всего на 25% по сравнению с выполнением одной такой задачи в DOS. То есть вы можете одновременно запустить форматирование диска, упаковку данных в Rbase, компиляцию в интегрированной среде Turbo C (!), копирование с другого дисковода на винчестер и при этом работать в любой другой задаче (например в текстовом редакторе) и не замечать загруженности машины.

Запуск DOS программ под Windows

Почти все программное обеспечение DOS полностью поддерживается Windows. При запуске DOS задачи система выделяет объем оперативной памяти либо требуемый программой, либо в соответствии с параметрами, содержащимися в PIF (Program Information File) этой программы. (Такие файлы входят в комплекты всех современных программных продуктов или создаются в PIF редакторе). Затем загружаются командный процессор DOS и сама программа.

На компьютерах с процессорами 386/486 DOS задачи можно отображать не на полном экране (опция Full screen), а в окно (опция Windowed) кроме тех DOS программ, которые адресуются прямо в видеопамять.

Вы можете просто запустить в системе Command.com и работать в DOS с любой программой, не ощущая присутствия Windows ни по скорости выполнения (при небольшом количестве выполняемых задач), ни по обращению к периферийным устройствам.



Программное обеспечение под Windows

Только за один год после выпуска MS Windows 3.0 Microsoft увеличила свои доходы на 60%. Свидетельством окончательной победы на рынке программных продуктов стало признание основного конкурента Microsoft фирмы Borland International, которая выпустила новые версии своих компиляторов C++ и Pascal 6.0 для Windows. Новые версии таких известных программных пакетов, как Norton Utilities 6.0, Lotus 1-2-3, также вышли с поддержкой работы в Windows. Широкие возжности Windows открыли перспективы не только для создания новых программных продуктов в области баз данных, электронных таблиц, гипертекстов и высококачественной графики, но и позволили программам, существующим в других операционных системах, работать на PC. Рассмотрим несколько основных групп программного обеспечения, предназначенного для работы в Microsoft Windows.

Электронные таблицы (Spreadsheet)

Microsoft Excel 3.0. Система с удобным и простым интерфейсом пользователя, возможностями обработки большого количества числовой информации, построения графиков, встроенным макроязыком, разносторонним обслуживанием печатающего устройства, COM порта, модема. Также существует поддержка дополнительных средств Windows, таких как динамический обмен данными (Dynamic Data Exchange – DDE) и объектно-ориентированная связь (Object Linking and Embedding – OLE). В системе предусмотрено создание сценариев вывода информации.

Lotus 1-2-3 for Windows. Всемирно признанные электронные таблицы фирмы Lotus, удачно дополненные мощными средствами Windows.

Текстовые редакторы

Кроме редактора, входящего в комплект поставки системы, с выпуском третьей версии появилось много текстовых процессоров для Windows. Приведем сведения о самом популярном из них.

Microsoft Word for Windows. Кардинально отличается от DOS версии удобной оболочкой, возможностью совмещать текстовую информацию с графикой, работает с разнообразными шрифтами, печатающими устройствами и.т.д.

Настольные издательские системы

Ventura Publisher фирмы Rank Xerox и PageMaker производства Aldus Plus. Для них Windows – необходимость. В отличие от DOS эта система может дать возможность с достаточной скоростью формировать документы любых размеров с большим количеством шрифтов и графической информацией, представленной в разных графических форматах.

Гипермедиа и гипертекстовые системы

Guide. Гипертекстовая система фирмы OWL International Inc. До появления Windows предназначалась для работы только на Macintosh. Позволяет быстро создавать гипертекстовые базы с текстовой и графической информацией.

ToolBook фирмы Asymetrix. Мощное средство для создания гипермедиа и гипертекстовых систем любой сложности. Система имеет объектно-ориентированный язык программирования для работы с графическими объ-

ектами, базами данных, фильмами, звуковой поддержкой и т.д. Все программы создаются в интерактивном режиме, поддерживаются такие функции Windows, как DDE, модемная связь.

Системы управления базами данных

Omnis 5 производства Blyth Software. Первоначально продукт работал только на Macintosh. Позволяет программировать базы данных в интерактивном режиме и с помощью утилиты автоматического программирования Express. Обрабатывает объем информации до 2560 Мбайт, имеет сетевую поддержку.

SuperBase. Позволяет при минимальных затратах труда программиста создавать мощные СУБД с несложными типами записей.

Paradox 3.0 – продукт Borland International. Всемирно известная система управления базами данных, существенно выигрывшая от перехода из DOS в Windows.

dbVista III для Windows – программный продукт Raima Corporation. СУБД для программистов – незаменимое средство в создании программного обеспечения, работающего со сложными базами данных.

Языки программирования

Microsoft C (версия 5.1 и выше), дополненный Software Development Kit – остается на сегодняшний день основным средством программирования Windows, так как в нем наиболее полно реализованы ее возможности.

Borland C++ фирмы Borland International. Вторая версия одного из самых популярных компиляторов Turbo C++, позволяющая как программировать на C++, так и отлаживать программы в Windows.

Turbo Pascal for Windows. Версия Pascal с набором объектов для Windows программ и с версией Turbo Debugger для Windows.

Сетевой адрес автора: tararaev@isrir.msk.su (сеть RELCOM) ■

Страсти по Windows

Мне хотелось бы высказать некоторые соображения по поводу очень важной и интересной темы, поднятой в статье Д.Тарараева "Microsoft Windows: новый подход в программировании".

На мой взляд, совершенствование аппаратных средств не было основной причиной, приведшей к появлению такой системы, как Windows – первая версия Windows появилась в 1986 году и была хилым ребенком в значительной степени из-за слабых, если можно так выразиться, "железных тылов". Не является такой причиной и требование обеспечить многозадачность – здесь как раз Windows проигрывает таким "натуральным" многозадачным системам, как UNIX и OS/2. Дело в том, что Windows базируется на однозадачной ОС MS DOS со всеми вытекающими отсюда, последствиями, и ее можно назвать многозадачной лишь отчасти.

По-видимому, главным фактором, сыгравшим роль в рождении систем типа Windows, была настоятельная необходимость провести очередную революцию в организации взаимодействия "человек-машина". Другой существенной причиной появления Windows явилось создание СТАНДАРТНОГО графического интерфейса. Здесь важны оба фактора.

Графический интерфейс позволяет общаться с компьютером, используя зрительные образы. Всем важным объектам, которыми оперирует пользователь (например, программам, дисководам, принтерам), ставится в соответствие маленькая открытка - пиктограмма. И вместо устаревшей стратегии "напиши и сделай" пользователь теперь имеет возможность действовать по принципу "покажи и сделай". Согласитесь, что гораздо легче иметь дело с символическими картинками, которые сразу же вызывают нужные ассоциации, чем запомнить малопонятные сокращения и названия. Например, если необходимо распечатать доклад, то достаточно с помощью "мышки" выбрать соответствующую ему картинку (если Вы не завели специальной картинки, то ее роль может сыграть просто имя файла), перенести ее по экрану и бросить над картинкой, изображающей принтер. И все. А в DOS пришлось бы набрать следующую строку:

```
copy c:/doc/doclad.txt prn
```

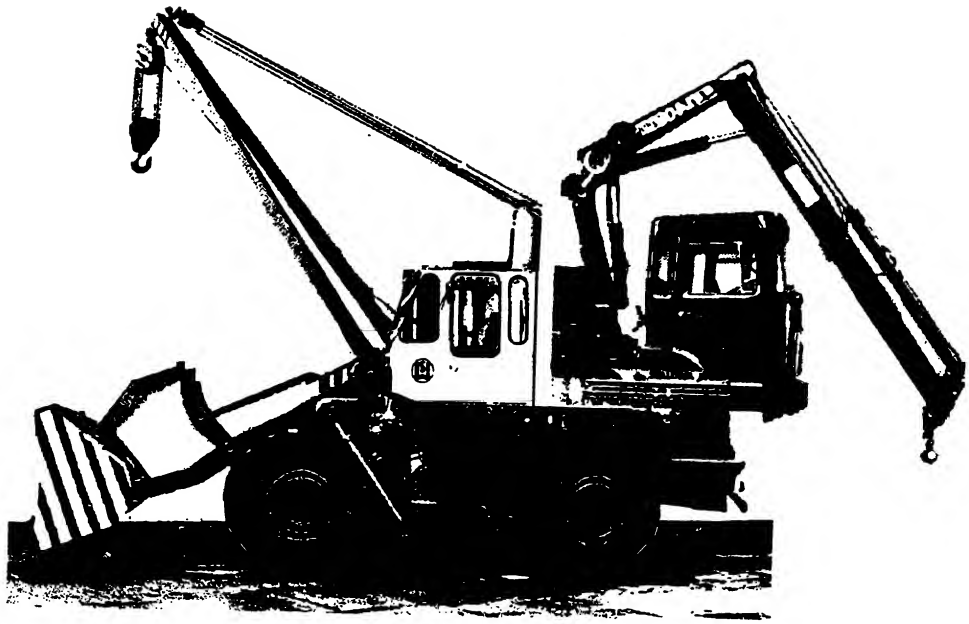
Графический интерфейс отличается от своего символьного предка большей гибкостью и утонченностью. Но это в целом положительное свойство таит в себе опасность: гибкость связана с разнообразием, а слишком большое разнообразие приводит к неразберихе. В принципе графический интерфейс не новость – он и раньше использовался в издательских пакетах, игрушках и т.д. Но каждая такая программа имела свой язык образов, в который было необходимо вживаться. Поэтому важна другая заслуга Windows – она обеспечивает стандартный интерфейс как при работе с самой системой, так и с приложениями, написанными для нее. Такая стандартизация имеет еще один важный аспект – внутренний. Так как программы, создаваемые для Windows, должны соблюдать определенные соглашения API (Application program interface), то между ними легко организовать взаимодействие, например, обмен сообщениями и данными.

Подводя итог вышесказанному, можно выделить три основные свойства Windows: наглядность интерфейса, связь свойств и действий с конкретными объектами (локализация) и взаимосвязанное выполнение нескольких приложений. Как известно, "ничто не ново под этим солнцем". Поэтому хотелось бы кратко упомянуть о тех истоках, благодаря которым стал возможен такой прекрасный результат. Основы визуального интерфейса были заложены в новаторских работах исследовательского центра Palo Alto фирмы Xerox (PARC). Результаты исследований были реализованы в 1981 году на 8010 Star Information System и позже на персональном компьютере Lisa в 1983 году.

Второе свойство тесно связано с другой пионерской разработкой PARC - созданием концепции объектно-ориентированного программирования. Это означает объединение в одном объекте данных и функций для оперирования этими данными (например, окно, меню и т.д.). Одной из первых практических реализаций этой методологии было создание языка SmallTalk–80.

Третье свойство прекрасно себя зарекомендовало в так называемых интегрированных пакетах, когда под одной оболочкой объединяются несколько разных программ. В этом случае историю рождения идеи проследить труднее. По-видимому, попытки создания таких пакетов

стр. 10 ▶



Clipper 5.0 – новые возможности

Часть I

А. Чернышков

От транслятора для dBase к развитой системе программирования

Впервые программный продукт с наименованием Clipper появился в 1984 году. Целью этой разработки фирмы Nantucket Corporation было создание компилятора для языка системы управления базами данных dBase III фирмы Ashton-Tate. СУБД dBase III и ее дальнейшие модификации предоставляют пользователю простые средства для создания и обработки файлов реляционной структуры, то есть организованных в виде двумерных таблиц со столбцами-полями и строками-записями. Каждый файл базы данных может быть загружен для работы с ним в одну из десяти так называемых рабочих областей. Перемещение указателя с записи на запись возможно только в активной рабочей области, в качестве которой может быть выбрана любая из этих десяти областей, но всегда только одна. Пользователь имеет возможность менять структуру данных, связывать, объединять, разделять или изменять целые файлы. Язык СУБД dBase III представляет собой набор команд и функций, которые выполняются при вводе в командную строку в интерактивном режиме работы. С помощью этих же команд и функций составляются прикладные программы для выполнения в пакетном режиме работы. Можно разрабатывать процедуру выполнения прикладных программ в мельчайших деталях, создавать свои меню, задавать формы расположения информации на экране дисплея и в выходных распечатках. Прикладные программы помещаются в текстовые файлы и выполняются в режиме интерпретации под управлением резидентной части СУБД dBase III. Такая организация СУБД позволяет начинающему пользователю персонального компьютера (ПК) освоить основы обработки данных и программирования за сравни-

тельно короткое время, и именно это обстоятельство способствовало быстрому росту популярности dBase III. Однако при разработке больших программ пользователь неизбежно сталкивается с долгим ожиданием выполнения программы вследствие потерь времени на интерпретацию. Поэтому естественным дополнением к любой интерпретирующей системе должен быть компилятор программ в машинный код. В силу этой органической потребности фирма Nantucket Corporation оказалась не единственным разработчиком компилятора для языка dBase. Однако наибольшую популярность приобрел именно Clipper.

programmer

С нетерпением ждали в нашей стране появления новой версии компилятора Clipper – Clipper 5.0

Что же позволило этому компилятору занять особое место ?

Язык dBase имеет строго ограниченный набор команд и функций. Команды dBase можно разделить на 4 основных класса:

- создания, уничтожения и организации доступа к файлам базы данных;
- манипулирования данными (изменение значений);
- ввода данных с клавиатуры и вывода их на дисплей или устройство печати;
- служебные.

Большинство команд имеет следующий общий вид:

<глагол> [<зона действия>] [<список выражений>]

[<условие выполнения>] [<ограничение>]

где <глагол> – ключевое слово, определяющее вид действия с данными , а другие компоненты, заключенные в угловые скобки, представляют собой необязательные опции, включающие в общем случае какие-либо ключевые слова и/или выражения. Например, команда

```
LIST ALL FIO, DOLG FOR DOLG > 0
WHILE OTDEL = "001"
```

означает выдачу из текущего файла базы данных на стандартное устройство вывода (дисплей), начиная с первой записи, списка значений полей FIO и DOLG для записей, где значение поля DOLG больше нуля, и до той поры, пока значение поля OTDEL равно "001". Функции dBase возвращают значения, являющиеся результатом их работы. Часть функций требует передачи им параметра, которым может также являться результат выполнения другой функции. Например, после выполнения выражения

```
CurrMonat = CMONTH(DATE())
```

переменная CurrMonat будет содержать наименование текущего месяца. Для более полного ознакомления с набором команд и функций языка dBase читатель может обратиться к [1] и [2]. Для приложений с несложной обработкой базы данных относительно простой структуры этого набора команд и функций оказывается вполне достаточно. Однако для разработки более сложных проектов на профессиональном уровне

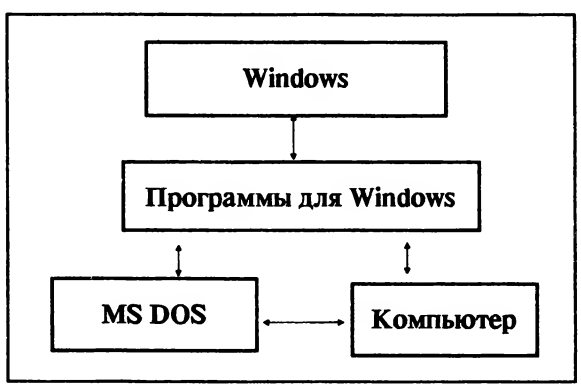
начало на стр. 8

предпринимались еще на "больших" машинах. Что же касается персональных компьютеров, то безусловно стоит упомянуть пакет Lotus 1-2-3 как возможно не самую первую, но самую успешную реализацию.

Хотелось бы также коснуться терминологии. На мой взгляд, не совсем корректно называть Windows операционной системой. Под операционной системой обычно понимают некую замкнутую структуру, обеспечивающую взаимодействие программ - приложений и аппаратных средств. Windows же сама является DOS программой и при своей работе отчасти использует средства MS DOS. Связь между программой для Windows, самой Windows и MS DOS упрощенно показана на рисунке.

Так как же все-таки называть Windows? Фирма Microsoft предлагает название "графическая среда". Учитывая, что здесь нет еще устоявшейся русской терминологии, я рискну предложить название "графическая операционная среда".

В заключение хочу добавить немного о новых приложениях для Windows. В конце концов именно они определяют силу Windows и эффективность его использования. И только глядя на них, пользователь может ответить для себя на вопрос:"А зачем мне нужна Windows?"



Взаимодействие различных компонент при выполнении программы под Windows

Приложения

Norton DeckTop for Windows - прекрасная оболочка для Windows, устраняющая недостатки, присущие Program и File Manager. Она представляет также множество полезных утилит как известных по пакетам Norton Commander, Norton Utilities, Norton Backup, так и новых.

Текстовые редакторы Describe Word Processor, Lotus Amni Pro, NBI Legacy.

Графические пакеты Adobe Streamline, Art&Letters, CorelDraw, Publisher's PaintBrush 2.0 и т.д.

Рынок приложений для Windows насчитывает уже сотни названий. А количество проданных экземпляров самой Windows минувшим летом увеличилось в 2 раза по сравнению с предыдущим годом. Триумфальное шествие продолжается!

П. Емельянов ■

появляется необходимость расширения языка. Clipper с самого начала был задуман как язык с открытой архитектурой, важнейшей чертой которой является возможность написания собственных функций. Причем в функции можно передавать параметры как по значению, так и по ссылке, что позволяет возвратить более одного результата выполнения функции. Функции могут быть написаны с использованием команд и функций языка Clipper (dBase), а также других языков. Для связи с программами, написанными на других языках, имеется так называемая система расширения языка, представляющая собой библиотеку программ, обеспечивающих взаимную передачу параметров и преобразование представлений данных.

Язык dBase позволяет пользователю манипулировать пятью типами данных, из которых числа, символы, даты и логические величины могут быть как полями файлов базы данных, так и переменными оперативной памяти. Особый тип данных "поле примечаний" можно завести только в файле базы данных. Поле примечаний является по сути ссылкой на место нахождения произвольного текста, относящегося к каждой конкретной записи в другом файле, имеющем расширение DBT. Каждый файл базы данных в dBase мог иметь только одно поле примечаний. Clipper позволил иметь несколько полей примечаний на один файл. Незаслуженно обойденный в dBase тип данных "массив" восстановлен в правах. В одном массиве можно использовать элементы различных типов, кроме типа "массив", что позволяет создавать простейшие структуры данных. Функции работы с массивами (вставить элемент, удалить элемент, сортировать массив, искать элемент с заданным значением) облегчили создание программных конструкций типа "список", "очередь", "стек". Неудобство, состоящее в том, что массивы допускались только одномерными, можно было обойти, используя оператор "макроподстановки" (&), также усовершенствованный в Clipper. Стало возможным использование вложенных макроподстановок.

Например:

```
One = "&Two"
Two = "Three"
Three = "hello"
? &One          && Результат: "hello"
```

В макроподстановках можно использовать выражения и функции:

```
ExpMonat = "CMONTH(DATE())"
CurrMonat = &ExpMonat
```

В результате выполнения этих операторов переменная CurrMonat будет содержать наименование текущего месяца. Эти возможности позволяют выносить небольшие части программ в файлы для обеспечения простоты настройки программ на конкретные условия применения, без изменения текстов программ и перекомпиляции. Иллюстрация создания двумерного массива и функция доступа к отдельному элементу для Clipper Summer'87 приведены в примере 1.

Пример 1

```
*
*   Программа : ARRAY.PRG
*   Версия : Clipper Summer '87
*
*   Программа иллюстрирует создание псевдодвумерного массива
*   и доступ к его элементам
*
*   Компиляция и компоновка :
*       Clipper Summer '87 :   clipper array -l -m
*                              plink86 fl array lib extend

*PROCEDURE ARRAY
PRIVATE Arr[2],Arr1[3],Arr2[3]

Arr[1] = "Arr1"
Arr[2] = "Arr2"

Arr1[1] = "Элемент11"
Arr1[2] = "Элемент12"
Arr1[3] = "Элемент13"

Arr2[1] = "Элемент21"
Arr2[2] = "Элемент22"
Arr2[3] = "Элемент23"

? ElemArr2(Arr,1,2)                    && Результат "Элемент12"
? ElemArr2(Arr,2,2)                    && Результат "Элемент22"

RETURN

*   Функция, возвращающая значение элемента массива
*   и/или назначающая новое значение

FUNCTION ElemArr2
PARAMETERS      ArrayOne,;
                        I,;
                        J,;
                NewValue
PRIVATE _ArrayTwo,Element
_ArrayTwo = Arr[I]
Element = &_ArrayTwo[J]
if PCOUNT() > 3
    &_ArrayTwo[J] = NewValue
endif
RETURN Element
```

Дополнительные возможности Clipper предоставил введением в язык ряда функций работы с обычными символьными файлами, освободив тем самым от необходимости пользоваться только файлами с реляционной организацией, которая в некоторых случаях оказывается неэкономной по использованию дисковой памяти. Эти функции представляют собой минимально необходимый набор операций для работы с файлами (создание и уничтожение, открытие и закрытие, перемещение указателя, чтение и запись).

Количество одновременно открытых файлов возросло с 15 до 255. При разработке систем большой информационной сложности нормализация базы данных неизбежно приводит к увеличению количества файлов, с которыми необходимо работать одновременно. Например, если при работе с каким-либо документом (файлом), имеющим индексный файл, требуется использование информации из семи справочников (файлов), также имеющих индексные файлы, то количество одновременно открытых файлов должно быть 8 * 2 = 16 . Таким образом, в dBase потребовалось бы при программировании постоянно отслеживать, какие файлы открыты для работы и на чье место можно загрузить потребовавшиеся файлы. При работе в Clipper можно сразу открыть все необходимые файлы, что позволяет также повысить быстродействие за счет уменьшения потерь времени на лишние открытия и закрытия файлов.

Очень удобными оказались функции сохранения и восстановления отдельных элементов текущей среды. Концепция разработки программ типа "черного ящика" требует большой тщательности в контроле за тем, чтобы после возврата из функции в вызвавшую ее программу последняя не получила бы непредвиденных изменений среды, таких например, как изменение цвета выводимых на экран данных, изменение активной рабочей области, а также установленных фильтров и связей в файлах баз данных и т.п. Строго говоря, структура большинства функций должна быть следующей:

- сохранить среду;
- выполнить необходимые действия;
- восстановить среду.

В Clipper имеются функции для запоминания и восстановления активной рабочей области, номера текущего индекса и индексного выражения, выражения фильтра в активной рабочей области, изображения участка экрана, текущего цвета, используемого при выводе на экран, и ряда других элементов среды.

Обеспечена возможность проверки вводимых данных при полноэкранном редактировании списка данных в функциях, разработанных пользователем.

И наконец ряд новых функций (ACHOICE(), DBEDIT(), MEMOEDIT()) позволяет быстро создавать высококачественные интерфейсы пользователя. Так, функция ACHOICE() организует работу меню, элементы которого задаются в массиве. Как параметры на вход функции задаются координаты экрана для вывода меню, начальная позиция для установки курсора, допустимость выбора той или иной позиции меню. Функция MEMOEDIT() является по сути простейшим текстовым редактором, который в умелых руках программиста может стать инструментом для создания редакторов, не намного уступающих по набору функций профессиональным. Функция DBEDIT() позволяет в определенном окне экрана просматривать и корректировать записи одной или нескольких рабочих областей. На первый взгляд эта функция является подобием функции BROWSE() языка dBase, однако возможности ее гораздо выше. Все эти три функции объединяет возможность передачи им в качестве параметра имени функции, которую программист может написать сам с целью обработки клавиш, нажимаемых пользователем. Таким образом, можно запрограммировать реакцию на нажатие "горячих клавиш" в зависимости от текущего режима работы функции и организовывать интерфейс с пользователем на любой вкус. В примере 2 представлена программа создания и редактирования базы данных, иллюстрирующая некоторые возможности функции DBEDIT().

Все описанные новшества, введенные в Clipper, позволили фирме Nantucket Corporation занять нишу между процедурными языками высокого уровня типа Pascal и интерпретирующими СУБД. Версия Clipper Summer '87 по праву стала считаться вполне самостоятельной системой программирования, и все больше программистов переходят к программированию сразу на языке Clipper вместо рекомендованной ранее технологии отладки программ в режиме интерпретации СУБД dBase и последующей обработки компилятором Clipper. Популяризации Clipper способствовали организованные фирмой Nantucket центры поддержки в крупных городах мира, где пользователи всегда могли получить консультации по телефону, а также выпуск периодического журнала "Nantucket News", распространяемого среди зарегистрированных пользователей. Популярность этого языка привела к тому, что ряд независимых фирм рискнул разработать коммерческие программные продукты для расширения возможностей языка. Большим спросом, например, пользовалась библиотека программ Clipper Tools One, в которой представлены функции работы с окнами, функции расширенного драйвера с аппаратурой IBM PC , функции коммуникации через последовательный интерфейс, новые функции обработки строк, функции преобразования чисел и операций над разрядами, новые видеофункции, функции для работы с гибкими и жесткими дисками, новые функции работы с печатающим устройством, новые функции преобразования даты/времени и др.

Однако, несмотря на популярность, Clipper Summer '87 имеет ряд недостатков, которые требовали дальнейшего развития системы. Основным

Пример 2

```
*
*  Программа        :   NEW_BASE.PRG
*  Версия           :   Clipper Summer '87, Clipper 5.01
*
*  Программа иллюстрирует некоторые возможности языка по созданию
*  и работе с файлами базы данных
*
*  Компиляция и компоновка :
*      Clipper Summer '87 :  clipper new_base -l -m
*                            plink86 fi new_base lib extend
*
*  Clipper 5.01              clipper new_base /l/m
*                            rtlink fi new_base /pil:base50


*  Процедура создания новой базы данных

*PROCEDURE NEW_BASE

   CLEAR
   READEXIT(.T.)
   CREAT New_stru
   USE New_stru
   PRIVATE NameFlds[4],HeadFlds[4]
   NameFlds[1] = "FIELD_NAME"
   HeadFlds[1] = "Имя поля"
   NameFlds[2] = "FIELD_TYPE"
   HeadFlds[2] = "Тип"
   NameFlds[3] = "FIELD_LEN"
   HeadFlds[3] = "Длина"
   NameFlds[4] = "FIELD_DEC"
   HeadFlds[4] = "Точность"
                 * Ввод имен полей новой базы
   @ 3,14,23,66 BOX " ╔╗ ║ ╝═╚ ║╲ "
   DBEDIT(4,15,22,65,NameFlds,"UserFunc","",HeadFlds)

   RELEASE NameFlds

   DELETE ALL FOR EMPTY(Field_name)   && Удаление
                                      && пустых записей
   PACK
   GO TOP
   IsStru = .NOT.EOF()
   USE
   IF IsStru

      CREAT NewFile FROM New_stru     && Создание новой
                                      && базы данных
      CLEAR
      PRIVATE NameFlds[FCOUNT()]   && Обьявление мас-
                                   && сива имен полей
      AFIELDS(NameFlds)   && Заполнение массива
                          && имен полей
      KEYBOARD CHR(0)
      INKEY()

      DBEDIT(4,0,22,79,NameFlds,"UserFunc")    && Редактиро-
                                   && вание и ввод записей
   ENDIF

   QUIT

   RETURN

*  Функция обработки нажатия клавиш при работе в DBEDIT()

FUNCTION UserFunc
   PARAMETERS mode,;        && Режим состояния после
                            && нажатия клавиши
              fld_ptr       && Номер поля в массиве,
                            && переданном в DBEDIT()
   PRIVATE cur_field,nKey
   nKey = LASTKEY()
   cur_field = NameFlds[fld_ptr]

DO CASE

   CASE mode = 1     && Нажата клавиша
                     && управления курсором
         IF  EOF()
             GO BOTTOM
             RETURN 2      && "Освежить" записи на экране
         ENDIF

   CASE mode = 2     && Попытка установки курсора за
                     && последней записью
         IF .NOT.EOF()
             RETURN 3      && Отобразить на экране
                           && добавление записи
         ENDIF

   CASE nKey = 27          && Нажата клавиша Esc
         RETURN 0          && Завершить работу DBEDIT()
   OTHERWISE
      IF ( nKey > 48 .AND. nKey < 127 ) .OR. nKey = 13
         IF ( nKey > 48 .AND. nKey < 127 )
             KEYBOARD CHR(nKEY)
         ENDIF
         IF EOF()
             APPEND BLANK  && Добавление записи
         ENDIF
         SET CURSOR ON
         @ ROW(),COL() GET &cur_field VALID Val1()
                                  && Редактирование поля
         READ
         SET CURSOR OFF
   IF LASTKEY() = 13 .OR. (LASTKEY() > 31 .AND. LASTKEY() < 127)
             IF (fld_ptr = FCOUNT())
                 KEYBOARD CHR(29) + CHR(24)
             ELSE
                 KEYBOARD CHR(4)
             ENDIF
         ENDIF
         RETURN 1              && Продолжить работу
      ENDIF

   ENDCASE
   RETURN 1

*  Функция проверки валидности вводимых полей данных
FUNCTION Val1
   PRIVATE Ret,NameVar
   Ret = .T.
   NameVar = READVAR()
   IF (NameVar = "FIELD_TYPE")
      Ret = (UPPER(&NameVar) $ "CNDLM")
   ENDIF
   RETURN Ret
```

■

из них, на мой взгляд, является следующий. Использование всего богатства языка позволяет в относительно короткий срок разрабатывать сложные программы, однако по мере увеличения объема программ им становится недостаточно оперативной памяти стандартного размера в 640 Кбайт. Создание оверлеев требует от программиста большей тщательности в проработке иерархии модулей. Да и оверлейные структуры не всегда спасают, т.к. возрастающие в процессе эксплуатации программы требования как правило приводят к добавлению новых модулей, и, таким образом, над программистом всегда довлеет боязнь наткнуться на нехватку памяти. Имеющаяся в системе программа управления запуском исполняемых программ SWITCH.EXE требует специальной организации передачи данных из одной исполняемой программы в другую, которая в свою очередь затрудняет модификацию программ. К тому же разбиение одного исполняемого модуля на два приводит к почти двухкратному увеличению дисковой памяти, используемой для хранения программ. Некоторые пользователи хотели бы иметь в системе словарь данных, сетевую совместимость блокировок файлов/записей с системой dBase и FoxBase, версию для системы UNIX, а также интерпретирующий режим для разработки и отладки.

И вот осенью 1989 года президент фирмы Nantucket Ларри Хаймендингер сделал официальное сообщение о том, что скоро появится система Clipper 5.0 , в которой будет решена проблема управления памятью. Система будет иметь препроцессор и поддерживать определяемые пользователем команды (User-Defined Commands). Новые локальные и статические переменные будут исключать возможность конфликта имен и упрощать создание функций типа "черного ящика". Станет возможным использование многомерных массивов. Интерпретатора в системе Clipper 5.0 не будет, но будут разработаны развитые средства отладки.

Эту версию у нас в стране ждали с нетерпением. Однако появившийся только летом 1990 года Clipper 5.0 поначалу разочаровал программистов. Часть отлаженных в Clipper Summer '87 программ отказывались сразу работать после трансляции в Clipper 5.0 . Великолепный по замыслу интерактивный отладчик программ иногда давал сбои. Таким образом, работа с новой версией приносила больше мучений в поисках ответа на вопрос "Почему не работает то, что работало раньше?", чем удовлетворение новыми возможностями. И только следующая версия Clipper 5.01, в которой были устранены имевшиеся ошибки, позволила спокойно приступить к изучению всех нововведений.

Итак, что же нового появилось в Clipper 5.0 ?

Уточнена и расширена классификация памяти для переменных. Наряду с имевшимися классами памяти FIELD (поля базы данных) и MEMVAR (переменные памяти) введены два новых класса:

• LOCAL (локальные переменные);

• STATIC (статические переменные).

Введены три новых типа данных:

• блоки кода;

• константа NIL;

• объекты.

Блоки кода позволяют осуществить доступ к скомпилированной программе, как к данным. Эта возможность аналогична макроопределению, но быстрее работает и более эффективна. Как и другие величины, значения блоков кода могут быть присвоены переменным, переданы как параметры и возвращены функциями. Значение константы NIL автоматически присваивается всем неинициализированным переменным (кроме описанных как PUBLIC).Если при вызове процедуры или функции некоторые параметры опускаются, то вместо них передается константа NIL. С типом данных "объект" связано введенное в Clipper новое понятие – класс, являющееся одним из главных понятий объектно-ориентированного метода программирования. В Clipper 5.0 введены 4 класса объектов, из которых на первых порах наибольший интерес может представить TBrowse Class, позволяющий с минимальными затратами организовывать просмотр на экране и редактирование табличных данных, а также создавать программы типа электронных таблиц. Программист может создать один или несколько экземпляров объектов того или иного класса. Каждый созданный экземпляр будет иметь тип "объект".

Расширены возможности работы с массивами.Элементы массива могут теперь содержать ссылки на другие массивы, что позволяет создавать многомерные массивы и сложные структуры данных. Предоставляются возможности динамически изменять размерности массивов, обеспечивая этим простоту создания списков и очередей. Массивы можно создавать и инициализировать литералами.

Компилятор Clipper 5.0 включает в себя полный препроцессор, позволяющий во время компиляции присоединять файлы программ и описаний, компилировать псевдофункции, задавать препроцессорные константы, проводить условную компиляцию, а также изменять существующие и определять собственные команды.

Новый отладчик Clipper Debugger позволяет отлаживать программу, имея на экране ее исходный текст. Кроме того, он обладает и рядом других возможностей по отладке программ.

Точки прерывания вызывают остановку выполнения всякий раз, когда они встречаются в исходной программе.

Точки просмотра позволяют контролировать значения переменных или выражений.

Точки трассировки вызывают остановку выполнения всякий раз, когда изменяется значение определенной переменной или выражения.

Окно стека вызовов позволяет наблюдать работу программы и всех незаконченных процедур (функций).

Обеспечивается возмож-

ность проверки и изменения переменных в незавершенных процедурах.

Процесс отладки становится очень похожим на отладку программ в таких системах как Turbo Pascal или Turbo C.

И наконец новый компоновщик RTLink позволяет программисту почти не задумываться о том, что оперативная память имеет предел 640 Кбайт. RTLink сам разобьет программу на динамические оверлеи и обеспечит их своевременный вызов в оперативную память на место неиспользуемых в этот момент других оверлеев.

С более подробным изложением новых возможностей Clipper 5.0 мы познакомим читателя в следующих номерах журнала. ■

Литература

1. Крамм Р. Системы управления базами данных dBase II и dBase III для персональных компьютеров. М.: Финансы и статистика, 1989.
2. Стогний А.А. и др. Программное обеспечение персональных ЭВМ. К.: Наукова думка, 1989.

Передовые технологии 90–х годов

Концепция объектно-ориентированного программирования

Основная борьба на рынке персональных компьютеров (ПК) разворачивается в сфере создания программного обеспечения. Пока еще не ясно, какая фирма первой создаст программное обеспечение следующего поколения. Microsoft пытается дополнить новыми возможностями систему Windows. Сейчас Windows продается, как вспомогательное средство для того, чтобы сделать программные продукты, использующие MS DOS, более похожими по виду и дружелюбию к пользователю на программное обеспечение для компьютера Macintosh фирмы Apple.

В 1992 году Microsoft обещает вытеснить MS DOS новой операционной мультизадачной системой,использующей RISC микропроцессор. IBM и Apple предпочли более радикальный подход и собираются совместно разработать принципиально новую операционную систему,основанную на принципах объектно-ориентированного программирования. Так что союз между IBM и Apple является не только предвестником новой ситуации в индустрии персональных компьютеров, но и началом превращения объектно-ориентированного программирования (ООП) из малоизвестной методики в главное средство, помогающее справиться с быстро растущей сложностью разработки программного обеспечения.

ООП является своего рода библией для одного из ведущих разработчиков программного обеспечения – фирмы Borland.

Как утверждает вице-президент фирмы Джэн Вэнг, тот факт,что всего за 18 месяцев фирма выпустила 3 новых версии Quattro Pro, можно полностью отнести на счет преимуществ использования ООП. Как утверждают официальные представители Borland, сейчас каждый программный продукт этой фирмы создается в соответствии с технологией ООП. Не следует однако думать, что концепция ООП является панацеей . Для того чтобы эта технология давала должную отдачу, сначала требуется создать обширную библиотеку объектов, что требует значительных финансовых затрат,а также времени. По этим причинам данный подход может стать преобладающим лишь к концу 90-х годов.

Базы данных для IBM–совместимых ПК

Продукт	Цена однопользовательской версии (ф.ст)	Цена run-time (ф.ст.)	Цена LAN-версии (ф.ст.)	Версия	Фирма	Число служащих	Оборот последнего года	Оборот предпоследнего года
Advanced Revelation	695	150	495	2.02	Revelation Technologies	120	L10 млн.	$7 млн.
Clarion	679	–	–	2.07	Dark Star	185	$130 млн.	–
Clipper	595	–	–	5.0	Nantucket	200	–	–
DataEase	595	–	695	4.2	Sapphire	200	L15 млн.	L10.5 млн.
DataFlex	895	200	800	3.0	Dataflex	35	L1.75 млн.	L1.25 млн.
DataPerfect	425	–	–	2.1	WordPerfect	2750	$452 млн.	$281.4 млн.
dBase IV	595	–	–	1.1	Ashton Tate	1430	$265.3 млн.	$307.3 млн.
Delta V	595		1095	II	Compost	30	L1.2 млн.	–
FormBase	495	–	–	1.1	Venture	–	–	–
FoxPro	645	500	250	1.02	Fox Software	200	–	–
Paradox	595	–	895	3.5	Borland	850	$200 млн.	$113 млн.
Progress	1650	313	3000	6.0	Progress	330	$40 млн.	$25 млн.
Reflex	250	–	–	2.0	Borland	850	$200 млн.	$113 млн.
Retieve	165	–	–	4.5	Sage	148	L13 млн.	L9 млн.
SmartWare	695	–	795	1.5	Informix	1000	$145 млн.	$104 млн.
SuperBase	595	–	895	1.2	Precision Soft	55	$5.1 млн.	$2.7 млн.

$ – амер. доллар, L – фунт стерлингов

Источник: PC Databases, an Evaluation and Comparison, ButlerBloor Ltd., April, 1991

PC : история и сегодняшний день

По мнению специалистов в области персональных компьютеров (ПК), обьявленный 3 июля 1991 года фирмами IBM и Apple совместный проект по созданию ПК нового поколения знаменует собой конец почти десятилетней эры,начавшейся с появления на рынке 12 августа 1981 года первого ПК фирмы IBM, который установил фактический стандарт в этой области на много лет вперед. Давайте кратко рассмотрим основные события в мире ПК за прошедшее десятилетие. Немного истории ...

1981 год

Через четыре года после появления первого ПК фирмы Apple,IBM вышла на рынок со своим IBM PC,который начал свое победное шествие по всему миру.

1982 год

Фирма Compaq, основанная Родом Кэньэном, приступила к производству портативного аналога IBM PC и стала самым крупным конкурентом фирмы IBM в области производства ПК .

1983 год

Фирма Lotus Development,возглавляемая Митчи Капором,выпустила программу Lotus для работы с электронными таблицами по цене 340 $,которая очень быстро стала наиболее успешно продаваемой программой для работы с IBM PC . IBM пытается протолкнуть на рынок ПК для дома и для школ свою неудачную модель PCjr.

1984 год

Фирма Apple торжественно обьявляет о создании ПК Macintosh с удобными графическими возможностями . Один из основателей фирмы Стив Джобс называет Macintosh "безумно удачной" моделью. IBM начинает выпуск PC AT - важной модификации ее первоначальной модели PC .

1985 год

Одновременно с падением объема продажи компьютера Macintosh Стив Джобс был уволен из фирмы Apple. IBM прекращает выпуск PCjr.

1986 год

IBM начинает выпуск своей первой лэптоп-модели, совместимой со стандартом PC . Модель была не очень удачной . Более успешной оказалась лэптоп-модель фирмы Toshiba, которая также использовала стандарт PC .

1987 год

IBM заменяет свою первоначальную модель PC новой PS/2 . Архитектура новой модели труднее поддается имитации по сравнению с PC . Тем самым IBM надеется ослабить позиции компаний,копирующих ее разработки . Microsoft совместно с IBM объявляют о создании операционной системы OS/2, которая имеет графические возможности, сходные с возможностями компьютера Macintosh. По их утверждению,эта система вытеснит первую операционную систему для ПК MS-DOS.

1988 год

Обьем продажи лэптопов фирмы IBM начинает расти. В продаже появляется разновидность меньших по размеру персональных компьютеров типа записной книжки.

1989 год

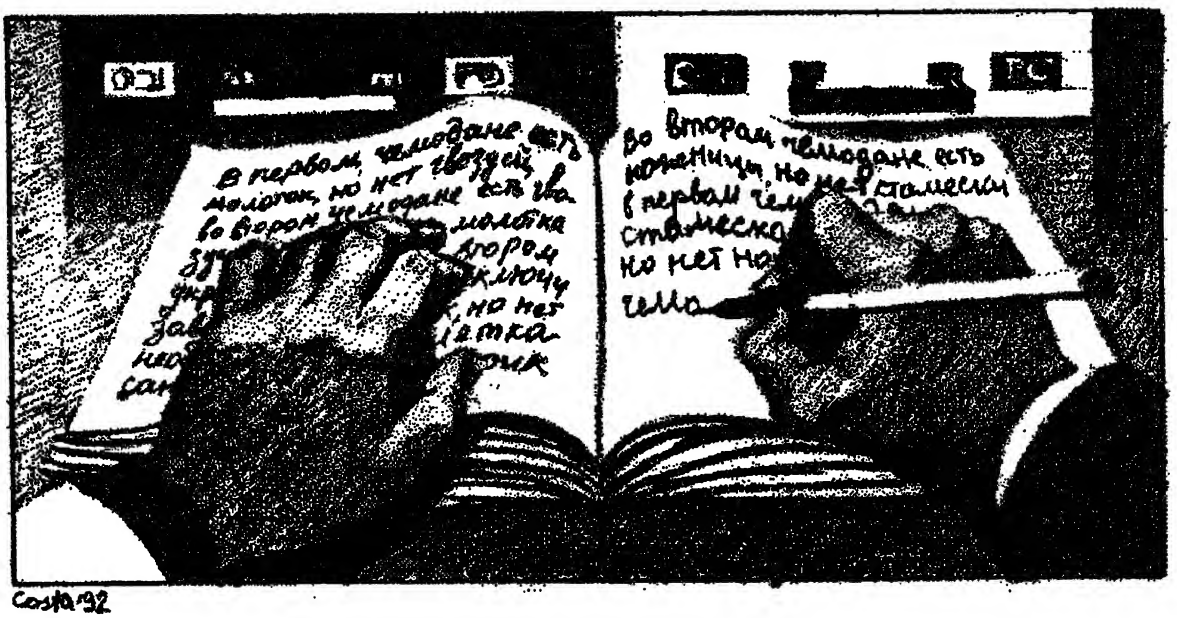
Появляется програмное обеспечение нового типа - "мультимедиа" для компьютеров, совмещающих возможности работы с видеоизображениями,звуком и текстом. Президент Microsoft Билл Гейтс предсказывает большое будущее этому направлению.

1990 год

Длившееся 10 лет сотрудничество между IBM и Microsoft начинает распадаться, после того как Microsoft стала проталкивать на рынок свою графическую программу Windows за счет OS/2.

1991 год

Фирмы IBM и Apple объявили о намерении работать вместе над созданием ПК нового поколения. Годовой объем продажи ПК во всем мире в 1991 году достиг уровня 93,3 миллиарда долларов по сравнению с 8,5 миллиардами в 1981 году , в США – 37,1 миллиардов долларов (в 1981 году – 3,9 миллиарда).



Идеи в программировании не становятся популярными без программ, на них основанных. И если речь идет о языках программирования, то для них нужны компиляторы. Объектно-ориентированных языков сейчас уже много, но самые известные – это Турбо Паскаль 5.5 и Си++. Познакомимся же с ними!

Сравнительное жизнеописание двух чемоданов с инструментами

М. Черкашин

Принято рекламировать языки следующим образом : "Этот язык – новое слово в компьютерной технике; он кардинально отличается от всего ранее существовавшего и т.д.". На практике это означает, что программисту следует забыть то, что он знал раньше; давно привычные вещи теперь ему придется делать совершенно новым, более прогрессивным способом. Не удивительно, что большинство программистов относятся к новым языкам программирования без энтузиазма. К счастью, на Си++ и Паскаль 5.5 это не распространяется. Конструкции, появившиеся в этих языках, создают новые возможности, удобные для программистов. Но то, что Вам нравилось в языках Си и Паскаль, тоже осталось. Поэтому приступайте к их новым версиям без боязни! Если Вы знаете Паскаль – хорошо! Если Вы знаете еще и Си – тем лучше! Но даже если Вы их и не знаете, а все равно возьметесь за эту статью, это будет просто замечательно.

В дальнейшем для краткости будем называть Турбо Паскаль 5.5 ТП 5.5.

Нашим КОЗЛОМ ОТПУ-

ЩЕНИЯ будут неупорядоченные наборы целых чисел, состоящие из чисел между 0 и 63. В самом простом случае его представление такое:

```
struct set {
    int numbers [64];
}
```

Если в наборе число i отсутствует, то значение numbers [i] равно нулю, в противном случае – единице.

ОБЪЕКТЫ И КЛАССЫ. Абстракция данных в языках программирования возникла много лет назад. Так, в языке Паскаль она присутствует уже в развитом виде. С точки зрения машины данные представляют собой некую область памяти ЭВМ. В этой области хранится информация, подлежащая переработке программами. Нас интересует классификация данных с точки зрения применимых к ним операций. Если одна порция данных (значение) может быть подставлена вместо другой в любом контексте, то говорят, что эти значения являются экземплярами одного типа. Так, значения 2 и 2.0 имеют различный тип, поскольку в следующих конструкциях

```
int b [2];
int b [2.0];
```

первая является описанием массива b из двух элементов, а вторая просто некорректна.

Таким образом, тип – это то, что объединяет похожие объекты между собой. Однако "дверь может быть настежь не открытой, но и не запертой" – похожие объекты могут быть несовместимы в каких-то отдельных операциях. В традиционных языках все объекты, создаваемые пользователем, либо одного типа, либо полностью несовместимы.

Рассмотрим простой пример. Можно написать (Си, Си++)

```
struct Точка {
    int X,Y;
};

struct Рамка {
    int X,Y;
    ...
};
```

А процедуры в старом Си пишутся отдельно, и лучшее, что мы можем себе позволить – поместить в тексте программы эти процедуры поближе к описанию данных и хорошо откомментировать свои намерения. И все.

Что нам дает Си++? Рассмотрим следующий текст:

```
struct Фигура {
    int X,Y;
    void Зажечь();
    void Погасить();
};

struct Точка : public Фигура
{};

struct Рамка : public Фигура
{...}
```

Сразу становится ясно, что Точка и Рамка – частные случаи Фигуры. Кроме того, "стороннему наблюдателю", т.е. процедурам, не принадлежащим классам Точка и Рамка соответственно, доступны только те данные и процедуры, которые расположены после ключевого слова public. Описание процедуры выносится отдельно. Однако воспользоваться этими процедурами не по назначению нам уже не удастся – компьютер поймает нас за руку.

До сих пор использование классов было вопросом удобства. Однако наследование предоставляет большие возможности. Легко написать программу, изображающую на экране движущуюся точку. Для этого достаточно последовательно зажигать и гасить ее в промежуточных состояниях между начальным и конечным положением точки.

Легко переписать процедуру движения для любой фигуры – но очень нудно. Кроме того, Вы можете смеяться, но в этом случае приходится для каждой процедуры выдумывать новое имя, а эта проблема подчас вызывает головную боль. Лучше написать одну процедуру на все случаи жизни. Вот так:

```
void Двигать (int dx, int dy, int n)
{
    int i;

    for (i=0; i<n ; i = i+1) {
        Зажечь ();
        Погасить ();
        }
}
```

Структура класса Фигура при этом должна быть заменена следующим образом:

```
struct Фигура {
    ... /* как в предыдущем
примере */
virtual void Зажечь();
virtual void Погасить();
void Двигать (int, int, int);
};
```

Правда перед этим необходимо оповестить машину, что процедуры Зажечь и Погасить нужно искать в классе, относящемся к конкретной фигуре. Так, во фразе "Собираясь на работу, никогда не забывайте закрывать за собой двери" процедуру "закрывать двери" нужно снабжать описателем virtual. Двери-то бывают разные – с висячим замком, с кодом.

Использование дружественных функций, полиморфизма, конструкторов и деструкторов показано в примере.

ДРУЖЕСТВЕННЫЕ ФУНКЦИИ. Функции, принадлежащие классу, вызываются с вполне определенным синтаксисом:

```
имяобъекта.процедура (параметры);
```

Это удобно, это подчеркивает связь процедуры с объектом. Иногда все же получается неуклюже. Например, странно выглядит конструкция x.sin. А математики определяют синус для многих объектов, включая матрицы. Некоторые виды функций и эти объекты вполне могут быть включены в иерархию классов.

Поэтому иногда оказывается удобным сохранить традиционный способ вызова процедуры. Для этого в Си++ введен механизм дружественных функций. Функция, снабженная описателем friend, имеет доступ ко всем данным и процедурам класса, а вызывается

Пример. Реализация множеств на Си++

```
struct set {
    int numbers [64];
    static No=0;
    public:
    friend int power (set);
    friend operator! (Point);
    friend set operator+ (set, set);
    void set (int);
    void ^set ();
}

int power (set cl)
{
    int i, co;

    for (i=0; i < 64; i++)
        if (numbers [i] == 1)
            co += 1;
    return (co);
}

set set::operator! (set cl)
{
    int i;
    set work;

    for (i=0; i<63; i++)
        if (cl.numbers [i] == 1)
            work.numbers [i] = 0;
        else
            work.numbers [i] = 1;
    return (work);
}

set set::operator+ (set set1, set set2)
{
    int i;
    set work;

    for (i=0; i<63; i++)
        if (set1.numbers [i] == 1 ||set2.numbers [i]==1)
            work.numbers [i] = 1;
        else
            work.numbers [i] = 0;
    return (work);
}

void set.set (int n)
{
    numbers [n]=1;
    No++;
    "No = " >> No >>"\n" >>out;
}

void set.^set ()
{
    No--;
    "No = " >> No >>"\n" >> out;
}

main ()
{
    set x;
    x = set (6) + set (7) + set (8);
    "Power : ">>
    power (x) >>
    power (!x) >>
    "\n"       >> out;
    "Power : ">>
    power (5) >>
    "\n"       >> out;
}
```

■

как обычная процедура. Итак, пишем (вспоминая нашего козла отпущения):

```
struct set {
    ...
    friend int power (set);
}

int power (set cl)
{
    int i, co;

    for (i=0; i<64; i=i+1)
        if (numbers [i] == 1)
            co += 1;
    return (co);
}
```

ПОЛИМОРФИЗМ. Существуют операции, которые напрашиваются на то, чтобы быть использованными с различными типами данных. Например, еще в старом Си или Паскале можно было написать

```
2*3
2.0*3.0
```

В данном примере операцией умножения обозначаются умножение целых и умножение вещественных чисел. А это, между прочим, две разные вещи.

В языке программирования (не только объектно-ориентированном) действия могут быть представлены процедурами либо операциями. В Си++ процедуры и операции (из списка уже существующих в языке) могут быть определены для классов, заданных пользователем. При этом одно и то же имя процедуры или знак операции могут обозначать несколько различных действий. Однако не возникает никакой путаницы – всегда употребляется та из процедур, у которой количество и типы аргументов совпадают с количеством и типом фактических параметров.

```
overloaded abs;
int abs (int);
int abs (char *);
```

Здесь abs (5) обозначает вызов первой функции, а abs ("Name") – второй.

Кроме явного указания на перегруженность функций, перегруженные функции могут возникать при наследовании.

```
struct Point {
    int X,Y;
    procedure Draw;
}
```

```
struct Circle : Point {
    int Radius;
    procedure Draw;
}

struct {
    int Size;
    procedure Draw;
}
```

Имя Draw может обозначать три разных процедуры (для разных классов). Здесь нет никакой необходимости в описателе overloaded.

> **programmer**
>
> То, что Вы уже умеете делать – делайте как обычно. В этом отношении Турбо Паскаль 5.5 и Си++ не отличаются от старых Паскаля и Си

Операции ведут себя во многом подобно функциям с одним или двумя параметрами. Поэтому нет ничего удивительного в том, что перегруженность операций также возможна. Если каждой операции приписать имя, то можно будет описывать операции, подобно процедурам, не меняя даже синтаксис. Но так как операций довольно много, такое приписывание может привести к путанице. Хорошо бы придумывать эти имена каким-нибудь регулярным способом, что и сделано в Си++. Для любой операции @ выполняемая ей процедура обозначается operator.

Это может быть дружественная функция с одним аргументом, если операция унарная, и двумя, если бинарная. Или же эта функция принадлежит классу. Для функции, принадлежащей классу, особым аргументом служит объект, к которому она применяется – он всегда указывается при вызове функции. Поэтому такая функция с именем operator@ обозначает унарную операцию, если описана без аргументов, и бинарную, если с одним аргументом. Унарную операцию, описанную таким образом, можно применять как префиксно, так и постфиксно. Так, операция,описанная

```
friend operator! (Point);
```

может применяться так – X! и так – !X.

Перегруженные операции могут обозначать какие угодно действия. Однако упростить себе жизнь программист может только, если обозначение операции будет естественным и применяться для наиболее частых операций.

```
friend set operator+ (set, set); /* внутри описания класса */
set set::operator+ (set set1, set set2) {
    int i;
    set work;

    for (i=0; i<63; i=i+1)
        if (set1.numbers [i] == 1 || set2.numbers [i]==1)
            work.numbers [i] = 1;
        else
            work.numbers [i] = 0;
    return (work);
}
```

Здесь операция + применяется для обозначения объединения множеств. Это удобно, потому что совпадает с традицией.

КОНСТРУКТОРЫ И ДЕСТРУКТОРЫ. Появляясь на свет, объекты часто нуждаются в некоторой предварительной подготовке. Чаще всего это инициализация самого объекта, но бывают нужны и другие действия. При уничтожении объекта тоже часто бывает необходимо что-то сделать.

```
struct set {
    void set (int);
    void ^set ();
}

void set.set (int n)
{
    numbers [n] = 1;
    число=число+1;
}

void set.^set ()
{
    число = число-1;
}
```

Если идентификатор функции совпадает с именем класса, то эта функция является конструктором и выполняется каждый раз в момент возникновения любого объекта этого класса. Если же в классе cl присутствует функция с именем ^cl, то она называется деструктором и выполняется в момент исчезновения объекта (вызов delete, конец блока).

Конструктор с одним аргументом (вроде объявленного здесь) также вводит преобразование типа аргумента к типу данного класса. Так set (6) – множество чисел, состоящее из одного элемента – 6. Это преобразование производится неявно, если вызывается процедура с типом параметра cl. Так, power (6) равно 1.

Немножко о Паскале

Паскаль 5.5 проще, чем Си++, и это большой плюс. По языку часто можно догадаться, какими принципами руководствовался его автор. Исходя из этого, можно сказать, что Паскаль 5.5 сознательно создавался, во-первых, как расширение старого Паскаля, а во-вторых, его авторы пытались достичь минимума новых конструкций.

НАСЛЕДОВАНИЕ. Начнем с примера

```
type Figure = object
        X,Y : integer;
        procedure Light;        virtual;
        procedure Extinguish; virtual;
end;

Point = object (Figure)
        procedure Light;        virtual;
        procedure Extinguish; virtual;
end;
```

Тут, собственно, и объяснять нечего. Сравните этот пример с аналогичным примером для Си++, приведенным ранее. То, что там написано, здесь переписано на ТП 5.5. Кто знает Паскаль, может убедиться, что даже и то поверхностное различие, которое мы видим в этих примерах, объясняется разницей между старыми Си и Паскалем. Объекты Паскаля 5.5 похожи на записи, а в Си++ – на структуры. А запись в Паскале и структура в Си – это одна и та же конструкция, только записываются

они немного по-разному. И это вообще единственная причина, по которой рассмотренные примеры отличаются друг от друга.

Имя в скобках после ключевого слова object обозначает класс-потомок. Единственная разница в конструкции virtual – и опять чисто внешняя – это то, что при каждом описании такой процедуры нужно каждый раз ставить ключевое слово virtual. Этим подчеркивается тот факт, что эта процедура остается virtual в любом классе-потомке. Но точно так же и в Си++, просто там компилятор сам запомнит, что какой-то потомок этой процедуры был описан как virtual.

Полиморфизм также исчерпывается минимально необходимым уровнем. Две процедуры с одним и тем же именем могут появляться в разных классах (либо один из этих классов – предок другого, либо у них общий предок). Этот вид полиморфизма не требует какого-то особого описания, поэтому описатель overloaded отсутствует.

Дружественных функций нет. Все члены объекта считаются public.

КОНСТРУКТОРЫ И ДЕСТРУКТОРЫ. Конструктором, также как и деструктором, может быть процедура с любым именем без параметра. Однако вызывать ее нужно явно. Предусмотрено сокращенное применение конструкторов вместе с оператором new и деструкторов с оператором dispose. Так, следующие конструкции

```
new (p, Init);
dispose (p, undo);
```

эквивалентны соответственно

```
new (p);
p.Init;
p.undo;
dispose (p);
```

Кроме того, использование конструкторов обязательно для объектов, класс которых или какой-то из его предков содержит процедуры с описателем virtual. Если конструктор на самом деле не нужен, необходимо описать конструктор с пустым телом.

```
constructor Init;
begin
end;
```

Этот конструктор производит некую инициализацию системных таблиц (ссылки на функции с описателем virtual), и в отсутствие такой инициализации происходит системная ошибка. Не блеск, конечно, но что делать - такой язык.

Сравнительное жизнеописание

Теперь в общих чертах сравним два рассмотренных нами языка. Это сравнение тем более уместно, что среди всех объектно-ориентированных языков они наиболее распространены и поэтому являются в известной степени соперниками. Оба языка несут в себе черты своих предков Паскаля и Си. И это порождает различие между ними. Поэтому будем сравнивать только объектно-ориентированные конструкции, поскольку они-то по крайней мере новы для обоих языков. С этой точки зрения Паскаль *5.5* и Си++ отличаются в двумя основными моментами.

Объектное подмножество Паскаля *5.5* невелико. Это значит, что если человек уже знает Паскаль, то выучить Паскаль *5.5* ему совсем просто – и это немало. Кроме того, чем меньше язык, тем меньше компилятор – а значит и ошибок в компиляторе тоже меньше. Использовать слишком большой язык – неудобно, можно легко запутаться в конструкциях.

Однако Си++ тоже не очень большой язык, хотя он больше, чем Паскаль *5.5*. Просто его автор не стремился уменьшить язык насколько возможно. Поэтому в Си++ вошли многие конструкции, которых в Паскале *5.5* нет. Если ставить во главу угла инкапсуляцию (возможность скрыть от пользователя технические детали), преимущества Си++ оказываются весьма заметными.

Вспомним, как мы реализовали множества. В Си++ есть все возможности, чтобы представить множество целых чисел просто множеством, а не массивом из нулей и единиц. И даже если нам придет в голову сменить внутреннее представление множеств, снаружи класса set никто ничего не заметит. Поэтому мы можем копаться внутри, не опасаясь за судьбу тех программ, которые этот класс используют. В Паскале *5.5* многих из этих возможностей нет. В частности, мы не можем объявить какие-то поля объекта, как его личную собственность (private), и поэтому нет гарантии, что какая-нибудь особо прыткая процедура не проникнет внутрь объекта.

Кроме того, стремление сделать язык удобным для реализации испортило конструкторы и деструкторы. Иногда пользователь вынужден их применить, даже если ему это и не нужно, чтобы облегчить компилятору работу (в тех случаях, когда в классе есть хотя бы одна процедура или функция, описанная как virtual). А это уже совсем никуда не годится. Пользователь не должен ничего знать о таких особенностях реализации, как системные таблицы! Кроме того, если конструктор необходимо вызывать явно, что же тогда от него остается? Обычная процедура, да еще необходимость делать то, что должен делать компилятор.

Вторая особенность Си++ – это то, что старый Си при создании Си++ подвергся коренной переработке. Появилась, например, строгая типизация. А ряд конструкций, сохранив отчасти форму, изменил стиль. Например, записи в Паскале *5.5* остались записями – такими же, как и в старом Паскале. А вот структуры в Си аналогичной конструкции – приобрели немало новых свойств. Теперь они являются частным случаем класса и, например, могут содержать процедуры. Даже если они выглядят как добропорядочные структуры из старого Си, это еще ничего не значит – у них могут еще появиться потомки.

Кроме того, Си++ – строго типизованный, чего в принципе не могло быть в Си. ■

Технология создания оконного интерфейса

С.Курковский

Окна? Это очень просто!

В 60-х и начале 70-х годов удобство работы пользователя игнорировалось большинством разработчиков систем. Большая ЭВМ окружалась кондиционерами и обслуживалась целой армией операторов. Иначе говоря, условия работы диктовала машина, а не наоборот.

С появлением персональных компьютеров поняли, что для эффективной работы системы не достаточно только того, чтобы аппаратура и программы обеспечивали правильные результаты – не менее важным фактором является удобство работы пользователя. Поэтому в настоящее время ни один программный продукт, лишенный более или менее удобного интерфейса "человек - компьютер", не может рассчитывать на успех.

Один из самых простых способов улучшения внешнего вида программы, дающий в то же время весьма неплохие результаты, – это создание исчезающих окон. Сообщение, выведенное в специально появившемся окне с рамкой и тенью, будет выглядеть куда приятнее, чем если бы оно просто появилось где-нибудь в углу экрана.

Существует несколько типов окон, среди них – стековые и слоеные. При использовании стековых окон доступным является только последнее открытое окно. Чтобы получить доступ к предыдущему окну, надо закрыть (уничтожить) последнее. Таким образом, стековые окна представляют собой структуру стека – LIFO (last in, first out). Слоеные окна отличаются большей гибкостью. При их использовании доступным может стать любое окно. Ограничимся рассмотрением стековых окон как более простых.

Рассмотрим алгоритм работы с текстовыми окнами (создание, динамическое размещение в памяти, управление и уничтожение).

В таблице приведены последовательность действий при манипулировании окном, а также имена используемых при

этом функций (кроме библиотечных).

На первом этапе (сохранение экрана) динамически выделяется память для запоминания той части экрана, которая окажется занятой открытым окном. Это необходимо для того, чтобы, закрыв окно, можно было восстановить экран в первоначальном виде.

На втором этапе (создание окна) определяются границы окна, рисуются рамка и тень, выводится заголовок.

Работа на третьем этапе (использование окна) полностью зависит от пользователя. В тексте приведены несколько функций, которые немного упростят его работу.

Функции для работы с окном

ЭТАПЫ	ИСПОЛЬЗУЕМЫЕ ФУНКЦИИ
Сохранение экрана	savewin()
Создание окна	openwin() gotoxywin() fillframe () wrchar() takechar()
Использование окна	fillwin() cursor()
Уничтожение окна	closewin()

Заключительный, четвертый этап (уничтожение окна) – самый простой. Окно уничтожается и экран восстанавливается в первоначальном виде.

Ниже приведено описание функций и структур, использующихся в системе поддержки и управления окнами.

Ни одна из приведенных функций не дублирует библиотечных функций Турбо Си. Некоторые функции в своей работе используют библиотечную функцию geninterrupt() (вызов системного прерывания) и псевдопеременные _AH, _AL, _BH, _BL и др. для задания значений регистров процессора. Эти функции используются в основном для манипулирования видеопамятью. Их можно модифицировать так, чтобы обращаться непосредственно к видеопамяти, но тогда потеряется вся та простота и наглядность, которые в полной мере представлены при работе с прерываниями.

Составим структуру, в которой будет храниться вся необходимая информация об окне (пример). Структура должна быть составлена таким образом, чтобы с ее помощью можно было бы образовать линейный список (рисунок).

Рассмотрим программную реализацию алгоритма работы с текстовыми окнами на языке Турбо Си.

1. Заголовочный файл TWINDOW.H (листинг 1). В нем содержатся необходимые описания структур, прототипы функций и макросы.

2. Функция gotoxywin() (листинг 2). Используется для позиционирования курсора. Отличается от библиотечной функции gotoxy() тем, что позиционирует курсор относительно левого верхнего угла монитора, а не текущего текстового окна. Использует прерывание 10h – видеосервис.

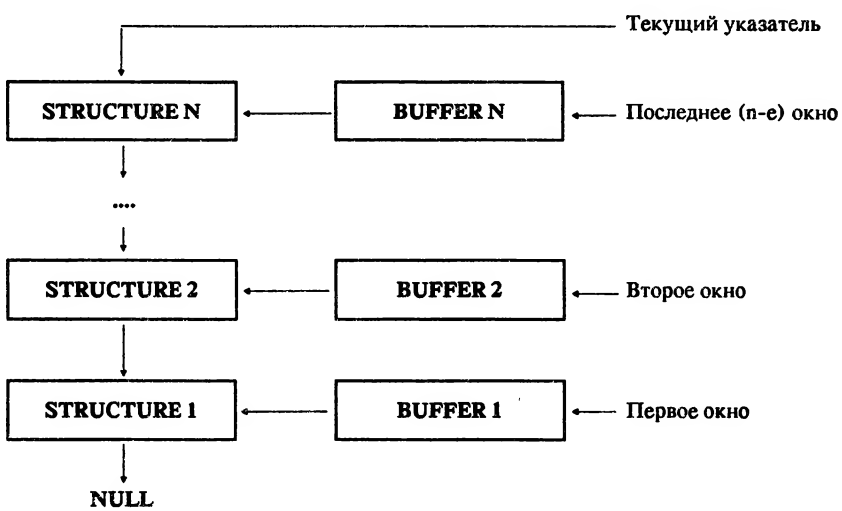
3. Функция wrchar() (листинг 3). Используется для вывода на экран одного или нескольких одинаковых символов без перемещения курсора. Использует прерывание 10h.

4. Функция takechar() (листинг 4). Возвращает символ, на который указывает курсор. Использует прерывание 10h.

5. Функция fillframe() (листинг 5). Используется для рисования рамки заданными сим-

Пример. Структура информации об окне

```
typedef struct w_status
{                   /* Переменные, служащие для хранения координат окна */
    int w_left,     /* Левый */
        w_top,      /* верхний угол */
        w_right,    /* Правый */
        w_bottom;   /* нижний угол */
    unsigned size;  /* Размер */
    void *buf;      /* и адрес буфера, в котором будет содержаться
                       видеоинформация */
    struct w_status *next;      /* Указатель на такую же структуру
                       позволит нам создать список, имеющий тип стека */
} WINDOW;
```



Линейный список структур хранения оконной информации

волами. Использует функции wrchar() и gotoxywin().

6. Функция copystr() (листинг 6). Данная функция копирует заданное количество символов, начиная с любого (также заданного), из одной строки в другую.

7. Функция cursor() (листинг 7). Используется для изменения формы курсора. В частности для того, чтобы сделать его невидимым, и для восстановления его предыдущих размеров. Использует прерывание 10h.

8. Функция fillwin() (листинг 8). Заполняет текстовое окно заданным символом с заданным атрибутом.

9. Функция getevent() (листинг 9). Является сильно упрощенным фрагментом функции, возвращающей сообщение о последнем событии. В данном случае возвращает скан-код нажатой клавиши.

10. Функция savewin() (листинг 10). При помощи этой функции происходит создание нового элемента стека, в котором будут записаны все характеристики окна и создан буфер, содержащий информацию непосредственно из видеопамяти. Этот буфер будет заполнен с помощью библиотечной функции gettext(). Таким образом, будет сохранена та часть экрана, которая окажется занятой новым окном.

11. Функция closewin() (листинг 11). Восстанавливает часть экрана, взятую из буфера с помощью библиотечной функции puttext(), уничтожает этот буфер и ненужную теперь структуру хранения оконной информации.

12. Функция openwin() (листинг 12). Создание окна. Если Вы сначала заглянули в листинг 12, Вам могло показаться, что эта функция имеет слишком сложный синтаксис. Отнюдь нет, там нет ничего лишнего! Функция openwin() проверяет, хватит ли свободной памяти для того, чтобы разместить новое окно. В случае, если работа прошла успешно, функция возвращает значение 1, если нет – 0.

13. Функция main() (листинг 13). Теперь осталось написать подходящую функцию main(), которая будет демонстрировать все достоинства приведенного алгоритма. Данная функция main() использует все вышеупомянутые функции, а пользователь с клавиатуры управляет работой программы : при нажатии клавиши "стрелка вверх" (на маленькой цифровой клавиатуре) создается новое окно, "стрелка вниз" – окно уничтожается, Esc – выход из программы. При этом координаты и цвета окон определяются с помощью генератора случайных чисел.

14. Проектный файл TWINDOW.PRJ пригодится для того, чтобы скомпилировать всю программу. Имена файлов приведены в соответствии с именами функций. Так подразумевается, например, что функция takechar() находится в файле с именем takechar.c, а функция main() находится в файле main.c (листинг 14).

programmer

Один из самых простых способов улучшения внешнего вида программы – создание исчезающих окон

Создание оконного интерфейса – дело не такое уж и сложное, как это может показаться. Время, потраченное на изучение этой статьи, с лихвой окупится – у Вас будет неплохой материал для создания красивых текстовых окон. Использование программных заготовок этой статьи позволит быстро и достаточно просто реализовать :

- создание и уничтожение текстовых окон;
- управление текстовыми окнами;
- управление курсором;
- доступ к видеопамяти. ■

Передовые технологии 90-х годов

Мультимедиа

"Multimedia computer" – это компьютер, совмещающий возможности телевидения и ПК в его первоначальном смысле, то есть компьютер многоцелевого назначения.

Вероятно, в комбинировании персональных компьютеров и бытовой электроники таятся наиболее многообещающие возможности.

Все указывает на то, что тенденция к интеграции ПК, видеомагнитофонов, стереоакустических цифровых систем и лазерных проигрывателей в одно целое будет усиливаться.

"Мультимедиа" может помочь индустрии ПК, находящейся в состоянии, близком к стагнации, обрести новые рынки сбыта своей продукции, включив в число ее покупателей потребителей бытовой электроники. Уже сейчас компании Commodore, Phillps, Tandy поставляют на рынок системы,в которых соединены цифровые проигрыватели компакт-дисков и персональные компьютеры.

Если сохранятся тенденции по постепенному превращению ПК в бытовую электронику в традиционном ее смысле,то японские компании смогут сильно подорвать позиции лидирующих американских компаний по производству компьютеров. Для потребителей бытовой электроники Sony - это фирма, которая несомненно предпочтительней IBM.

Пока что американские компании лидируют в производстве ПК и программного обеспечения, а японские – в производстве бытовой электроники. Нарастающая угроза со стороны японских компаний явилась одной из причин, обусловивших появление альянса IBM - Apple. Сделав фирму Apple своим партнером по созданию мультимедиа-компьютера, IBM надеется предотвратить установление более тесных связей между этой фирмой и японскими производителями электроники.

Листинг 1. Заголовочный файл TWINDOW.H

```
/*********************************
*    Заголовочный файл TWINDOW.H    *
*********************************/
#define DOUBL“ ╔╗╚╝╗ ”        /* Символы для */
#define SINGL “ ┌┐└┘ ”        /* рисования рамок */
#define MIX1 “ ┌┐└╝ ”        /* (функция fillframe) */

#define NSTR “”                /* Пустая строка */

/* Размер свободной памяти в зависимости от модели памяти ... */
/* ... в малых моделях ... */
#if defined(__TINY__) || defined(__SMALL__) || defined(__MEDIUM__)
#define MAXMEMSIZE coreleft() - 1000
/* ... в больших моделях ... */
#else
#define MAXMEMSIZE farcoreleft() - 1000
#endif
typedef struct w_status /* Структура для хранения */
{ /* информации об окне */
int w_left,
w_top,
w_right,
w_bottom;
unsigned size;
struct w_status *next;
void *buf;
} WINDOW;

enum events {noth = 1000,kbup,kbdown,kbesc};        /* Описание клавиш */

/*   Прототипы функций    */
char *copystr(char *dest, char *source, unsigned pos, unsigned count);
void cursor(unsigned on);
void gotoxywin(int x, int y);
void wrchar(int ch, int tattr, int count);
int takechar();
void fillframe(int x, int y, int width, int height,
                    int frameat, char *framechar);
void closewin();
void savewin(int x, int y, int width, int height);
int openwin(int x, int y, int width, int height,
                    int field, int text, int frame, int title,
                    char *titlestr, char *framechar, int shadow);
void fillwin(int ch, int tattr);
int getevent();
```

Листинг 2. Функция gotoxywin()

```
/*************************************
*    Функция позиционирования курсора    *
*************************************/
#include <dos.h>
void gotoxywin(x, y)
int x, y;    /* Задают координаты курсора относительно верхнего
             левого угла экрана (1,1) */
{
    _AH = 0x02;          /* Функция позиционирования курсора */
    _BH = 0;             /* Текущая видеостраница */
    _DH = y - 1;         /* Координаты курсора относительно */
    _DL = x - 1;         /* позиции (1,1) */
    geninterrupt(0x10);  /* Функция 10h - видеосервис) */
}
```

Листинг 3.Функция wrchar()

```
/*************************
* Функция печати символа *
*************************/
#include <dos.h>
void wrchar(ch, tattr, count)
int ch;     /* Символ, который будет напечатан */
int tattr;  /* Аттрибут символа ( Аттрибут = ЦветБуквы + 16 * ЦветФона ) */
int count;                  /* Счетчик - сколько раз печатать */
{
    _AH = 0x9;           /* Функция вывода символа на экран */
    _BH = 0;             /* Текущая видеостраница */
    _AL = ch;            /* Установить символ */
    _CX = count;         /* Установить счетчик */
    _BL = tattr;         /* Установить аттрибут */
    geninterrupt(0x10);  /* Функция DOS 0x10h - видеосервис */
}
```

Листинг 4. Функция takechar()

```
/*************************************
*    Функция возвращает символ из памяти    *
*************************************/
#include <dos.h>
int takechar()
{
    _AH = 0x08;        /* Функция извлечения символа из видеопамяти */
    _BH = 0;             /* Текущая видеостраница */
    geninterrupt(0x10);  /* Функция DOS 10h - видеосервис */
    return(_AL);         /* Возвратить символ */
}
```

Листинг 5. Функция fillframe()

```
/*************************************
*          Функция рисует рамку          *
*          заданного типа и цвета        *
*          в заданных координатах        *
*************************************/
#include <dos.h>
#include <conio.h>

void fillframe(x, y, width, height, frameat, frchars)

int x, y;                /* Координаты верхнего левого угла рамки */
int width, height;       /* Ее ширина и высота */
int frameat;             /* Аттрибут символов рамки */
char *frchars;           /* Строка символов для рисования рамки */
{
    int i;

    textattr(frameat);              /* Нарисовать верхнюю строку рамки */
    gotoxy(x, y);
    putch(frchars[0]);
    wrchar(frchars[1], frameat, width - 1);
    gotoxy(x + width - 1, y);
    putch(frchars[2]);              /* Нарисовать вертикальные границы */
    for(i = 1; i < = height - 1; i++)
    {
        gotoxy(x, y + i);
        putch(frchars[3]);
        gotoxy(x + width - 1, y + i);
        putch(frchars[7]);
    }

    gotoxy(x, y + height - 1);      /* Нарисовать верхнюю строку рамки */
    putch(frchars[4]);
    wrchar(frchars[5], frameat, width - 1);
    gotoxy(x + width - 1, y + height - 1);
    wrchar(frchars[6], frameat, 1);
}
```

Листинг 6. Функция copystr()

```
/*********************************
*          Функция копирует          *
*          count символов строки source,  *
*          начиная с pos, в строку dest   *
*********************************/
#include <string.h>
```

```
char *copystr(dest, source, pos, count)
char *dest, *source;
unsigned pos, count;
{
    int len;

    len = strlen(source);
    if( pos > len ) return(source);
    memcpy(dest, source + pos, count);
    strset(dest + count, '\0');
    return (dest);
}
```

Листинг7. Функция cursor()

```
/*************************************************
*       Функция делает курсор невидимым          *
*       или наоборот в зависимости от параметра on *
*************************************************/
#include <dos.h>

void cursor(on)
unsigned on;

{
    static int old_ch, old_cl;

    if( !on )
    {
        _AH = 0x03;
        _BH = 0x01;
        geninterrupt(0x10);     /* Узнать текущие характеристики курсора */
        old_ch = _CH;           /* и сохранить их для */
        old_cl = _CL;           /* последующего исрользования */
        _AH = 0x01;             /* Функция установки формы курсора */
        _CH = 0x20;             /* Убрать курсор */
    }
    else
    {
        _AH = 0x01;             /* Функция установки формы курсора */
        _CH = old_ch;           /* Восстановить характеристики
                                   /*формы курсора */
        _CL = old_cl;
    }
    geninterrupt(0x10);         /* Функция DOS 10h - видеосервис */
}
```

Листинг 8. Функция fillwin()

```
/*****************************************
*           Функция заполняет окно       *
*           символом ch с атрибутом tattr *
*****************************************/
#include <conio.h>
#include "TWINDOW.H"

void fillwin(ch, tattr)
int ch;
int tattr;

{
    struct text_info ti;
    int i;

    gettextinfo(&ti);           /* Узнать текущую видеоинформацию */
    for(i = ti.wintop; i <= ti.winbottom; i++)
    {                           /* Заполнить окно символом ch */
        gotoxywin(ti.winleft, i);
        wrchar(ch, tattr, ti.winright - ti.winleft + 1);
    }
}
```

Листинг 9. Функция getevent()

```
/*********************************
*   Функция возвращает информацию  *
*   о нажатых клавишах             *
*********************************/
#include <bios.h>
#include "TWINDOW.H"

int getevent()
{
    union           /* Структура для хранения скан-кода и символа */
    {   char ch[2];
        int i;
    } c;

    c.i = bioskey(0);           /* Считать скан-код */
    switch(c.ch[0])
    {
        case 0 :
            switch(c.ch[1])
                {
                    case 72 : return kbup;      /* Up    */
                    case 80 : return kbdown;    /* Down  */

                    default : return noth;
                };

        case 27 : return kbesc;                 /* Esc   */

        default : return noth;
    }
}
```

Листинг 10. Функция savewin()

```
/*****************************************
*       Функция сохраняет часть экрана,   *
*       занятую окном, и добавляет новый  *
*       элемент в список                  *
*****************************************/
#include <alloc.h>
#include <conio.h>
#include "TWINDOW.H"

extern long freememory;             /* Свободная память */
extern WINDOW *winstatus;           /* Указатель на стек */

void savewin(x, y, width, height)
int x, y, width, height;            /* Координаты и размеры окна */

{
    WINDOW *k;
    void *dest;
    unsigned s, s0;

    s0 = (width + 1) * (height + 1) * 2;    /* Размер буфера */
    s = sizeof(WINDOW) + 2;                 /* Размер структуры */
    k = (WINDOW *)malloc(s);                /* Создание нового элемента */
    dest = (void *)malloc(s0);              /* Создание нового буфера */
    freememory -= s0 + s; gettext(x, y, x + width, y + height, dest);
                                            /* Текст из видеопамяти - в буфер */
    k -> w_left = x;
    k -> w_top = y;
    k -> w_right = width + x;               /* Сохранение информации */
    k -> w_bottom = height + y;             /* в структуре WINDOW */
    k -> buf = dest;
    k -> next = winstatus;                  /* Новый элемент стека */
    k -> size = s + s0;
    winstatus = k;
}
```

Листинг 11. Функция closewin()

```
/*************************************
*       Функция уничтожает окно и     *
*       и восстанавливает часть экрана,*
*       занятую этим окном            *
*************************************/
#include <alloc.h>
#include <conio.h>
#include "TWINDOW.H"

extern long freememory;          /* Размер свободной памяти */
extern WINDOW *winstatus;        /* Указатель на стек */

void closewin()
{
    WINDOW *k;
    void *dest;

    if(winstatus != NULL)        /* Если список не пуст ... */
    {
        k = winstatus;
        dest = k -> buf;
            freememory += k -> size;
        puttext(k -> w_left, k -> w_top, k -> w_right, k -> w_bottom, k -> buf);
                        /* Текст из буфера - на экран */
        winstatus = winstatus -> next;
        free((void *)k);     /* Уничтожение последнего элемента списка */
        free(dest);          /* Уничтожение буфера */
        window (winstatus -> w_left + 2,  /* Восстановление предыдущих */
        winstatus -> w_top + 1,           /* координат окна */
        winstatus -> w_right - 4,
        winstatus -> w_bottom - 2);
    }
}
```

Листинг 12. Функция openwin()

```
/*******************************************
*           Функция создает новое          *
*       окно в зависимости от параметров   *
*       и добавляет новый элемент в стек   *
*******************************************/
#include <conio.h>
#include <string.h>
#include "TWINDOW.H"

extern long count;               /* Счетчик открытых окон */
extern long freememory;          /* Размер свободной памяти */

int openwin(x, y, width, height, field, text, frame, title,
        titlestr, framechars, shadow)
int x, y, width, height;         /* Координаты и размеры окна */
int field;          /* Цвета (а не аттрибуты) :  фона,        */
int text;                                  /*  текста,       */
int frame;                                 /*  рамки,        */
int title;                                 /*  заголовка     */
char *titlestr;          /* Указатель на строку - заголовок */
        /* если это указатель на пустую строку или NULL, */
        /* заголовок выведен не будет */
char *framechars;        /* Указатель на строку символов для рамки */
int shadow;              /* Нужно ли рисовать рамку */

{
    int i, titleat, fieldat, frameat;
    char tit[80];

    if(freememory > 5000)            /* Если хватит свободной памяти ... */
    {
        strcpy(tit, titlestr);
        fieldat = text + 16 * field;
        titleat = title + 16 * field;
        frameat = frame + 16 * field;
        savewin(x, y, width + 1, height);        /* Сохранить экран */
        if(shadow)                               /* Сделать тень */
        {
            for(i = y + 1; i <= y + height; i++)
            {
                gotoxywin(x + width, i);
                wrchar(takechar(), LIGHTGRAY + 16 * BLACK, 1);
                gotoxywin(x + width + 1, i);
                wrchar(takechar(), LIGHTGRAY + 16 * BLACK, 1);
            }
            for(i = x + 2; i <= x + width; i++)
            {
                gotoxywin(i, y + height);
                wrchar(takechar(), LIGHTGRAY + 16 * BLACK, 1);
            }
        }
        window(x, y , x + width - 1, y + height - 1);
        textattr(frameat);
        clrscr();                        /* Нарисовать рамку */
        fillframe(2, 1, width - 2, height, frameat, framechars);
        if (strcmp(titlestr, NSTR))      /* Если заголовок - */
        {                                /* не пустая строка ... */
            textattr(titleat);
            if (strlen(titlestr) width - 4)   /* Если он слишком длинный ... */
            {
                copystr(tit, titlestr, 0, width - 5);    /* ... обрезать его */
                strcat(tit," ");
            }
            gotoxy((width + 2 - strlen(tit)) / 2, 1);
            cputs(tit);                  /* Вывести заголовок */
        }
        window(x + 2, y + 1, x + width - 3, y + height - 2);
        textattr(fieldat);               /* Закрасить созданное окно */
        clrscr();                        /* нужным цветом */
        count++;
        return(1);
    }
    return(0);
}
```

Листинг 13. Функция main()

```
/*****************************************************
*       Функция демонстрирует возможности            *
*                работы с окнами.                    *
*       Управление : стрелка вверх - открыть окно,   *
*            стрелка вниз  - закрыть окно,           *
*                  Esc  -  выйти                     *
*****************************************************/
#include "TWINDOW.H"
#include <conio.h>
#include <stdlib.h>
#include <alloc.h>

int on = 1,off = 0;
WINDOW *winstatus;          /* Указатель на список оконной информации */
unsigned long count = 0;    /* Счетчик открытых окон */
unsigned long freememory;   /* Свободная память */

main()
{
    int x, y, width, height, FIELDCOLOR, TEXTCOLOR, BORDERCOLOR,
                TITLECOLOR;
    char *mix = MIX1;
    int mytextattr = BLACK + 16 * CYAN;

    freememory = MAXMEMSIZE - 5000;      /* Установить размер*/
                                         /*доступной памяти */
    winstatus = NULL;                    /* Начало списка */
    fillwin(' ', WHITE + 16 * BLUE);     /* Заполнить окно */
    cursor(off);                         /* Убрать курсор */
    textattr(mytextattr);
    gotoxy(1, 1);
    clreol();
    cputs("Демонстрация оконного интерфейса, написанного на Турбо Си");
    gotoxy(1, 25);
    clreol();
    cputs(" \x18 - Открыть окно     \x19 - Закрыть окно     Esc - Выход");

    for(;;)              /* Бесконечный цикл ... */
    {
        switch(getevent())  /* Получить информацию о нажатых клавишах */
        {
            case kbup :                  /* Создать новое окно */
```

```
        width = random(50) + 10;     /* со случайными */
                                     /* координатами. */
        height = random(15) + 4;
        x = random(79 - width) + 1; /* случайными размерами */
        y = random(23 - height) + 2;
        FIELDCOLOR = random(8); /* и случайными цветами */
        TEXTCOLOR = random(15);
        BORDERCOLOR = random(15);
        TITLECOLOR = random(15);
        if(openwin(x, y, width, height,
                FIELDCOLOR, TEXTCOLOR, BORDERCOLOR,
                TITLECOLOR, "Оконный интерфейс ", mix, on))
        {
                /* Если окно успешно создано ... */
            cprintf( "Окно # %u.\n\r", count);
            cprintf(" %lu байт памяти свободно.\n\r", freememory);
        }
                /* ... и если нет */
        else cprintf(    " Недостаточно памяти для создания
                         нового окна.\n\r");
        break;
    case kbdown : closewin(); break;    /* Закрыть окно */
    case kbesc :                        /* Выйти */
        window(1, 1, 80, 25);
                textattr(LIGHTGRAY + 16 * BLACK);
                clrscr();
                cursor(on);
                exit(0);                /* Выход из цикла */
            }
        }
    }
```

Листинг 14. Проектный файл TWINDOW.PRJ

```
main(TWINDOW.H)
openwin(TWINDOW.H)
closewin(TWINDOW.H)
savewin(TWINDOW.H)
getevent(TWINDOW.H)
gotoxywin
wrchar
takechar
cursor
fillwin
fillframe
```

Персональные компьютеры становятся товаром широкого потребления, а их производители вынуждены менять стратегию

Тезисы статьи из журнала "Economist" (2 - 8.11.1991)

Некогда символ технологического волшебства, персональные компьютеры стали товаром ширпотреба. Другие типы компьютеров движутся в том же направлении. Эти изменения сотрясают компьютерную индустрию до основания.

Делать хорошие компьютеры более недостаточно для успеха. Это может каждый.

С одной стороны, все большее количество элементов на кремниевых кристаллах приводит к уменьшению числа компонент на плате, и, следовательно, к сокращению простора для творчества производителей компьютеров.

С другой стороны, возросшая изощренность пользователей приводит их к требованию совместимости новых компьютеров с множеством старых, которые у них уже есть, что также ограничивает производителей в нововведениях.

Compaq постоянно возглавлял отрасль с одними из лучших ПК на рынке. Фирма продавала компьютеры через сеть дилеров, гордившихся своим умением приспособить основные машины к потребностям покупателей. Эта стратегия основывалась на общей в компьютерной индустрии вере в то, что корпоративным покупателям нужны "решения" - полные системы аппаратуры и программного обеспечения, которые решают проблемы бизнеса одним нажатием кнопки. Однако сегодня большинство покупателей не желает платить за компоновку таких систем дополнительную плату. Цена полной компьютерной системы упала до суммы цен ее частей.

Прибыли Compaq упали с 454 млн. долл. в первые 9 месяцев 1990 года до 65 млн. долл. за тот же период 1991 года; продажи снизились на 8 процентов – до 2,4 млрд. долл. Но Compaq не одинок. Во всей отрасли маленькие фирмы, которые сконцентрировались на том, чтобы делать немногие вещи превосходно, процветают за счет больших, которые принимают обычные разговоры о "решениях" всерьез и стараются делать хорошо много разных вещей. Как результат доля доходов 50 крупнейших компьютерных фирм резко упала.

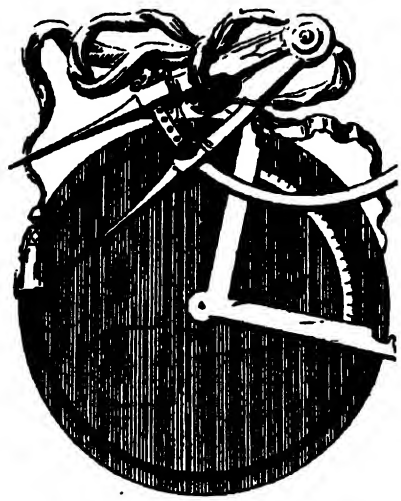
Процветают фирмы, которые позволяют другим делать ту работу, которую они не могут делать наилучшим образом. Microsoft сконцентрировалась на программном обеспечении; Dell, Northgate и другие пионеры продажи компьютеров по почте - на снижении стоимости и улучшении обслуживания. Японские фирмы процветают за счет высокого качества микросхем. Arthur Andersen и EDS - на построении систем из чужих аппаратуры и программ.

Вопреки распространенному убеждению прибыли производителей программ не превосходят прибылей производителей аппаратуры. 5 из 10 наиболее быстро растущих компьютерных компаний за 1985 - 1990 годы - аппаратные фирмы (Sun, DEC, Compaq, AST, Apple).

Чтобы преуспеть в проектировании программного обеспечения и компьютеров, нужны детальные знания о процессах производства электронных компонент. Поэтому Microsoft тесно взаимодействует с Intel; Sun– с Texas Instruments.

Программа распечатки имен текущих каталогов

В.Водолазкий



При отладке программ, которые активно переключают текущие каталоги и рабочие диски, необходимо получать информацию о том, какие каталоги в настоящий момент времени установлены по умолчанию. Вдоволь намучавшись с оболочками типа Norton Commander, я написал простую программу на ассемблере (см. листинг), которая выводит на дисплей имена рабочих каталогов для всех дисковых накопителей (ГМД, винчестеров, RAMдисков), определенных в системе. Для решения этой задачи оказалось очень удобно использовать одну из недокументированных функций DOS - 52h (получить адрес таблицы переменных DOS). При вызове этой функции в регистры ES:BX возвращается адрес таблицы переменных DOS, которая содержит много интересной информации.

В предлагаемой программе используются лишь два значения из этой таблицы: число дисководов в системе и адрес списка текущих каталогов. Следует отметить, что список каталогов в каждой из версий DOS имеет различный формат, а следовательно и длину. Мною были проведены исследования трех версий MS DOS, результаты которых приведены в таблице.

Поэтому, если Вы работаете с другой версией операционной системы, Вам может потребоваться внести изменения в программу. Каждая запись о текущем каталоге содержит полное имя каталога (вместе с идентификатором диска) в виде ASCIZ-строки и некоторую дополнительную информацию, которую, к сожалению, расшифровать не удалось.

Для сборки программы необходимы TASM 2.0+ или MASM 5.0+ и компоновщики LINK или TLINK. Командные строки имеют следующий вид:

```
TASM DEFDIR
TLINK DEFDIR
или
MASM DEFDIR DEFDIR NUL NUL
LINK DEFDIR;
```

■

Длины записей списка текущих каталогов для трех версий MS DOS

Версия	3.30	4.0	4.1
Длина записи	51h	58h	58h

Программа распечатки имен текущих каталогов

```
        name    Prog
        .radix  16
        page    60d,120d
;
;
;       В.В.Водолазкий
;
;       Программа распечатки имен текущих каталогов для всех
;       логических дисков. Версия для операционной системы MS DOS
;       3.30, 4.00, 4.01
;
;
;       03/11/91 12:07pm
;
;
;       .MODEL      SMALL

DRIVECNT    EQU     20H     ;количество дисков в системе
DefDir      equ     16h     ;адрес списка текущих каталогов

        .CODE

Start:
        mov     ax,@data
        mov     ds,ax
        MOV     AH,30           ; читаем версию DOS
        INT     21H
        SUB     AL,3            ; настраиваем индекс
        XOR     BX,BX
        MOV     BL,AL
        shl     bx,1
        MOV     AX, Word Ptr DELTA[BX]
        MOV     DDD,AX

        MOV     AH,52           ; читаем область данных DOS
        INT     21H

        xor     ch,ch
        MOV     Cl, byte ptr ES:DRIVECNT [BX]       ;читаем число дисков

        ; в CX - число логических дисков

        MOV     ax, WORD PTR ES:DEFDIR[BX]
        mov     dx, word ptr es:defdir[bx][2]

        push    es
        MOV     ES,DX           ;в es:dx - адрес данных о каталогах
        MOV     DX,AX

        ;
        ; формируем каталоги и ищем в них файлы
        ;
DIR_LOOP:
        cld
        push    es
        PUSH    DX
        PUSH    CX
        MOV     CX,65D
        mov DI,offset dir_name
        PUSH    DX
        POP     SI              ;si = dx
        PUSH    DS
        POP     AX              ;ax = ds (временно)
        PUSH    ES
        POP     DS              ;ds = es
        PUSH    AX
        POP     ES              ;es = ax ( = ds)
COPY_L:
        lodsb
        CMP     AL,0
        JE   COPY_END
        STOSB
        loop    copy_l
copy_end:
        MOV     CX,3

        PUSH    ES
        POP     DS
        LEA     SI,ADDR_CRLF
        REP     MOVSB
        mov     dx,offset dir_name
        MOV     AH,9
        INT     21

        POP     CX
        POP     DX
        pop     es
        add     dx, ddd         ;на следующую запись
        LOOP    DIR_LOOP
        POP     ES

;
; завершение работы программы
;
        PUSH    DS
        mov     ax,4c00
        int  21
;
        .DATA

defdef  dw  0,0

delta   dw  51,58h              ;длины записей для различных версий

DDD     DW ?

addr_crlf   db 0dh,0ah,'$'

KEY     DB  0

DIR_name    db   80d dup (0)

        .STACK  50
```

Внимание! Специалисты по компьютерной графике!

С 28 сентября по 2 октября 1992 года в Москве будет проходить вторая международная конференция по компьютерной графике - ГрафиКон'92. Конференция проводится под девизом "Компьютерная графика в науке и искусстве". Организаторы: Московское научно-техническое и творческое общество компьютерной графики "Графо", Институт прикладной математики им. М.В.Келдыша, Союз архитекторов, Институт физики высоких энергий (г.Протвино), при содействии американского общества компьютерной графики ACM SIGGRAPH и европейского общества Eurographics. Структура конференции ГрафиКон'92 будет напоминать структуру ежегодных конференций ACM SIGGRAPH (США).

Она включает:

- образовательные курсы по компьютерной графике
- приглашенные доклады
- конкурсные доклады
- выставка современных технологий компьютерной графики
- выставка компьютерного искусства
- компьютерный видеотеатр
- круглые столы и семинары.

Адрес Секретариата конференции: 125047, Москва, Миусская пл. 4, ИПМ, Оргкомитет ГрафиКон'92.
Телефон: 972-36-42, факс: 972-07-37.

Rambling Software Inc.

PC SHORT, version 0.18— сверхкомпактная (EXE-файл 40Кбайт) надстройка над ДОС с широким спектром возможностей, занимающая в ОЗУ при запуске программ менее 1,5 Кбайт.

PC SHORT включает:

- высокоэффективную файловую систему, поддерживающую всевозможные операции над файлами и каталогами (в том числе с нетривиальным способом проведения групповых файловых операций), и запуск исполняемых и batch-файлов;
- профессиональный текстовый редактор программиста с возможностью редактирования до 4 файлов в 10 окнах одновременно, со средствами изменения и записи конфигурации окон (активность, размер, цвет, приоритет, расположение), с развитыми возможностями строчного и блочного редактирования, поиска и замены, межоконных и межблочных операций;
- возможность временного выхода в ДОС с редактированием командной строки и прокруткой буфера набранных команд;
- калькулятор программиста с возможностями вычислений в десятичном, HEX и BIN форматах, сменой основания счисления и доступом из любой части системы (редактора, файловой подсистемы и т.д.);
- карту размещения резидентных программ в оперативной памяти;
- поддержку манипулятора "мышь", EGA/VGA адаптеров (режим 43/50 строк во всех средствах пакета), записную книжку, будильник и др.

PC SHORT отличается компактностью исполняемого кода, удобствами взаимосвязи файловой системы и редактора, естественностью и полнотой набора функций, продуманной эргономикой работы с клавиатурой и "мышью", минимально возможным в рамках ДОС размером резидентного модуля, высокой скоростью исполнения команд (для файловых операций на 20-100% выше, чем у известной оболочки Norton Commander 3.0) и способен эффективно работать в системах с отсутствием жесткого диска вследствие малого дискового пространства, занимаемого программой.

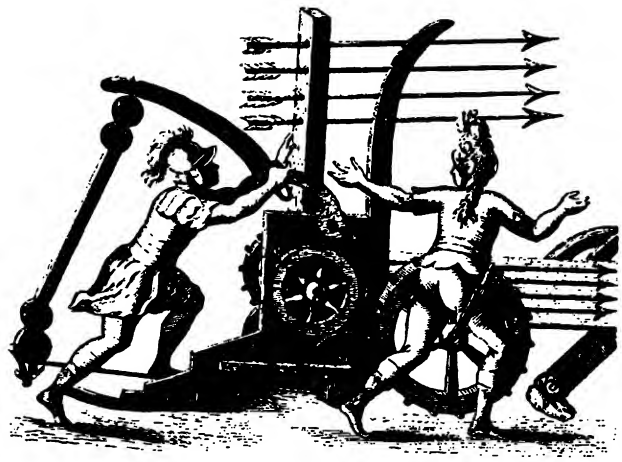
Данная версия программы предлагается для свободного распространения и является демонстрационной. Зарегистрированным пользователям гарантируются поставка последующих версий программы с 50-процентной скидкой и бесплатные телефонные консультации.

Сумма, необходимая для регистрации:
организациям —50 руб.
частным лицам — 25 руб.
Стоимость рабочей версии PC SHORT 1.20 — 550 руб.
По получении указанной суммы Вам будет выслано регистрационное удостоверение.

© 1991 НПО "ИНБЕЛ" © 1991 Rambling Software Inc.

Республика Беларусь, 210015, г.Витебск, ул. Ленина, 16-1, НПО "ИНБЕЛ",
р/с №70002 в ПИБ г.Витебска МФО №405036.
Тел. (8-0212)373809, (8-0212)339394 — для консультаций.

Бутовые вирусы на марше



В.Водолазкий

Обычно статья о компьютерных вирусах начинается с подробного изложения, кто это сделал и зачем, а заканчивается призывом "сурово наказать безответственных "технокрыс" (естественно, по просьбам трудящихся). Мы же постараемся отойти от шаблона и рассказать более подробно о работе бутовых вирусов (поражающих системный загрузчик – boot sector), и о том, как с ними бороться. В известной монографии Н.Н.Безрукова [1] утверждается, что бутовые вирусы постепенно вымирают. Но, к сожалению, не все оптимистичные прогнозы оправдываются.

Для более предметного обсуждения наш рассказ будет построен на одном из самых распространенных (на момент написания статьи) вирусе. Он получил условное название "6-е Марта", так как активизируется только в этот день, и мы надеемся, что данная публикация, во-первых, позволит предотвратить его проявление в массовых масштабах, а, во-вторых, поможет программисту в случае необходимости позаботиться о своей системе самому.

Этот вирус представляет собой "злобствующую" вариацию на тему M-05 (Марихуана) и свидетельствует о знакомстве его автора с передовыми достижениями австралийской вирусной технологии на 1988 год. Имеющиеся в распоряжении автора антивирусные средства: AIDSTEST (версия 138), –V (версия 170) не обнаруживают данный вирус, а VR (версия 90) просто зависает. Итак, рассмотрим устройство вируса.

Вирус "6-е Марта"

Вспомним, как происходит процесс загрузки с гибкого или жесткого дисков. Первый сектор нулевой дорожки диска 0/0/1 (далее мы будем кодировать номер сектора следующим образом: H/T/S – головка/дорожка/сектор) считывается в оперативную память ЭВМ по адресу 07C0:0000, и затем управление передается по этому же адресу. В обязанности программы, записанной в этом секторе, входит загрузка необходимого программного обеспечения (если диск содержит операционную систему) или выдача предупреждения об отсутствии системы на диске. С точки зрения бутового вируса это означает следующее:

- программный код вируса привязывается к конкретной области в адресном пространстве;
- вирус должен либо умещаться в объем 512 байт (длина сектора), либо обеспечивать последующую дозагрузку своего кода;
- для обеспечения маскировки присутствия вируса на диске необходимо сохранять оригинальный (первоначальный) загрузчик в каком-либо из секторов для загрузки на место вируса.

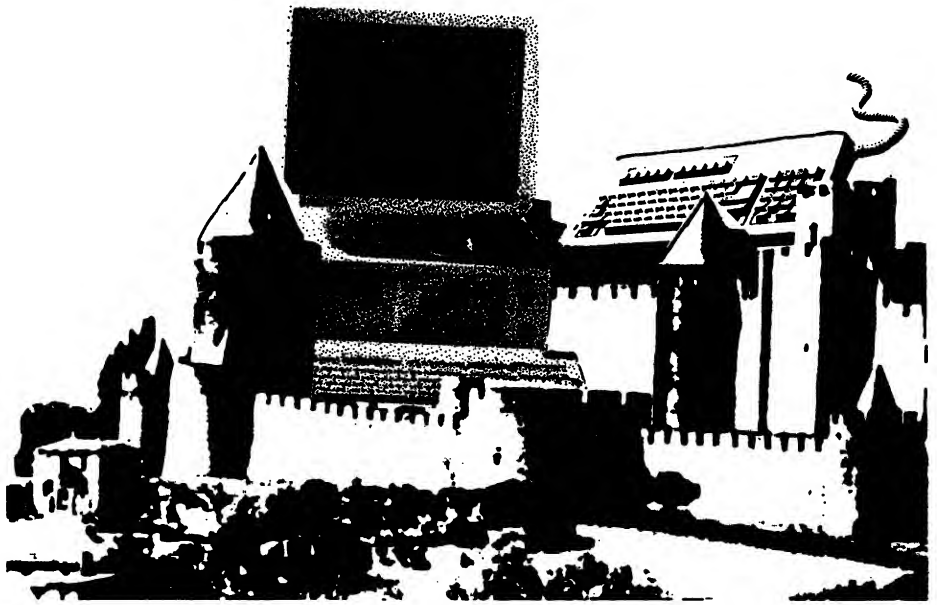
Как же решены эти проблемы в вирусе "6-е Марта"? Первая операция, выполняемая вирусом - это перехват вектора прерывания 13 - обслуживание дисковых накопителей и запоминание системного вектора в

теле вируса. После этого вирус уменьшаст объем доступной системе оперативной памяти на два килобайта:

```
xor     ax,ax
mov     ds,ax
.
.
.
mov     ax,ds:413h
dec     ax
dec     ax
mov     ds:413h,ax
```

Затем он вычисляет сегментный адрес начала освободившегося участка памяти и переписывает туда тело вируса. После этого управление передается копии. В результате активна копия вируса, размещенная в верхних адресах оперативной памяти, к которым для операционной системы уже нет доступа. Область памяти, закрепленная ROM BIOS за загрузчиком (07C0:0000 – 07C0:01FF), теперь свободна и может быть использована для активизации настоящего загрузчика.

Что же делает вирус дальше? Естественно, как и любое простейшее [2], он размножается. Вначале осуществляется проверка, заражена ли дискета в дисководе A. Если нет, вирус осуществляет заражение гибкого диска. Если же загрузка идет с гибкого диска, вирус пытается инфицировать первичный загрузчик (Master Boot Record — MBR) жесткого диска C. Это означает, что если вы пытаетесь загрузиться с дискеты, зараженной вирусом, то ваш винчестер немедленно (!) будет заражен. При этом совершенно не важно, есть ли на дискете операционная система, или в загрузочном секторе должна быть программа, сообщающая об отсутствии системы и требующая перезапуска. Такая "прожорливость" обеспечивает вирусу достаточно высокую скорость распространения и значительно затрудняет его ликвидацию (именно за счет огромного числа копий).



Идем дальше. Итак, вирус пытается инфицировать дискету или жесткий диск. После этого осуществляется считывание значения системного времени. Обратите внимание на реализацию:

```
    mov     ah,4
    int     1ah
    cmp     dx,306h ;это 6-ое марта
    je      time_has_come
    retf    ;передача управления
        ;оригинальному загрузчику
time_has_come:
    ;
    ; активизация целевой функции
    ; вируса
    ;
```

Дело в том, что приведенная последовательность команд будет корректно исполняться только для ЭВМ, совместимых с PC/AT. В PC/XT и наших советских пародиях типа ЕС1840 КМОП-таймера нет, и при включении ЭВМ системная дата и время сбрасываются. Это означает, что для вируса 6-е марта не наступит никогда!

Целевая функция вируса, получающая управление 6-го марта каждого года, состоит в разрушении информации на дисках. На это много ума не надо, и в комментариях данная часть кода не нуждается, а эпитеты читатель без труда подберет сам. Если же время для активизации вируса не пришло, управление передается настоящему загрузчику, спрятанному вирусом в одном из секторов, в зависимости от типа диска – носителя вируса (таблица).

После передачи управления загрузчику происходит штатная загрузка системы. При этом в оперативной памяти ЭВМ остается резидентная копия вируса, на которую подключен вектор 13. В примере 1 представлен обработчик вектора обслуживания дисковых накопителей, реализованный в вирусе "6-е Марта".

Как видно из программы, резидентная часть вируса осуществляет заражение только гибких дисков и только при обращении к дисководу. Процедура Replicate (не приведена) осуществляет заражение загрузочного сектора по следующему алгоритму:

Место хранения первоначального загрузчика вирусом "6-е Марта"

Тип диска	Тип носителя	Адрес сектора		
		Головка	Дорожка	Сектор
5.25" 1.2MB	F9	0	0	0E
5.25" 360KB	FD	0	0	3
НМД "Винчестер"	F8	1	0	7

• считывает загрузочный сектор дискеты во внутренний буфер в верхних областях оперативной памяти;

• по первым четырем байтам загрузчика определяет, имеется ли на дискете вирус;

• если вируса на дискете нет, то вычисляется номер сектора, в котором должен храниться оригинальный загрузочный сектор (см. таблицу);

• вычисленный номер сектора сохраняется в теле вируса;

• в сектор 0/0/1 производится запись тела вируса;

• осуществляется сохранение оригинального загрузчика.

Пример 1. Обработчик вектора 13h

```
Vir_Int13   Proc   Far
    push   ds
    push   ax
    or     dl,dl    ;доступ к ГМД?
    jnz    wint_service
    xor    ax,ax
    mov    ds,ax  ;настройка сегмента данных
                  ;на работу с переменными
                  ;BIOS
    test byte ptr ds:Disk_Motor_Run,1
    ; проверяем, включен ли двигатель A:
    jnz wint_service
    pop    ax
    pop    ds
    pushf
    call   cs:sys_vect_13    ;вызов истинного
                  ; обработчика вектора 13h,
                  ;сохраненного в теле вируса
    pushf
    call    Replicate        ;вызов процедуры,
                  ; обеспечивающей раз-
                  ;множение
    popf
    retf 2           ;вместо IRET
Wint_service:
    pushf
    call   cs:sys_vect_13
    retf   2
Vir_Int13   Endp
```

По окончании процедуры размножения управление возвращается программе пользователя.

Ну что же, теперь, когда алгоритм работы вируса более или менее ясен, давайте составим план, как нам "выжечь каленым железом" забравшуюся на диск "заразу".

programmer

Не так страшен вирус, как его малюют авторы антивирусных программ

Обнаружение и ликвидация вируса "6-е Марта"

Наиболее распространенный метод поиска вирусов – контекстный. Принцип его состоит в поиске определенных последовательностей байтов (т.н. сигнатур) в подозреваемых файлах (а в нашем случае – в загрузочном секторе). Выбор сигнатуры – дело тонкое и требует некоторого опыта. К счастью, автор вируса подумал об этом за нас, и мы можем использовать его решение. Как уже упоминалось выше, вирус определяет, заражена ли дискета (винчестер) путем проверки первых четырех байт загрузочного сектора. Если указанные байты имеют значение : F5 00 80 9E, то вирус считает, что диск уже инфицирован.

Поскольку вирусом не предусмотрено никаких фокусов для защиты от трассировки (перехват INT 09, подстановка спрятанного BOOT-сектора), для обнаружения вируса достаточно выполнить следующие операции:

• считать загрузочный сектор гибкого диска или MBR жесткого диска C;

• сравнить первые четыре байта считанного сектора с приведенным выше образцом (сигнатурой);

• если сигнатура совпадает, принять решение о наличии вируса.

Предположим, что проверка показала : диск заражен. Теперь нам необходимо предпринять ряд мер по ликвидации вируса.

Как правило, уничтожение вирусов на дисках осуществляется после перезагрузки с "чистой" дискеты. Этот метод следует признать единственно верным, но ведь "всяко бывает". Поэтому, прежде чем предпринять какие-либо активные меры, мы должны убедиться в том, что в оперативной памяти нет резидентной части вируса.

Для этого мы проверяем значение ячейки 0000:0413, в которой хранится объем доступной системе памяти. Это значение устанавливается в ходе отработки POST-кода, а затем корректируется вирусом. Для поиска вируса мы можем использовать следующий алгоритм:

1) вычисляем сегментный адрес области памяти, следующей за объявленной для системы;

2) если вычисленное значение указывает на видеоОЗУ ($A000), то вируса в памяти нет;

3) сканируем оперативную память, начиная с вычисленного сегментного адреса, по начало области видеоОЗУ;

4) если обнаружены байты сигнатуры, выдать предупреждение пользователю о наличии в памяти вируса.

Допустим теперь, что в оперативной памяти вируса нет. Мы можем переходить к проверке диска. Для этого мы считываем во внутренний буфер содержимое сектора загрузки (0/0/1) и проверяем его содержимое на предмет наличия той же самой сигнатуры. Если первые четыре байта содержимого буфера на совпадают с сигнатурой, можно не волноваться – вируса "6-е Марта" на данном диске нет. Если же вирус обнаружен, его необходимо удалить.

Ручное удаление вируса

Быстро и надежно! С помощью Norton Utilities. Автор настоятельно рекомендует пользоваться старой доброй версией 4.5. Фокус в том, что в DISKEDIT версии 5.0 есть ошибка, которая не позволяет выбрать ОДИН сектор. В результате у вас появляется реальный шанс развалить всю систему. Если не верите – попробуйте!

Для удаления вируса вам необходимо переписать сохраненный вирусом оригинальный загрузочный файл в загрузочный сектор. И все! Для определения места хранения загрузочного сектора вы должны воспользоваться таблицей.

Программное удаление вируса

Методика удаления вируса из программы в общем-то не меняется. Возникает только один вопрос. Если при ручном удалении вируса пользователь знает, какой диск стоит у него в дисководе, и может определить, откуда взять сектор, то как это может сделать программа? На это автор может ответить только следующим: "Но ведь вирус-то это может сделать...". Поэтому мы в очередной раз воспользуемся результатами труда автора вируса.

В примере 2 представлено начало вируса. Как видно из листинга, вирус сохраняет в своем теле адрес начального сектора. Поэтому, если вы взяли на себя смелость разработать собственный антивирус, то можете использовать эти данные для поиска оригинального сектора. Кстати, этот путь более надежен, чем попытки самому определить тип диска в дисководе. Ведь вирус мог ошибиться при установке типа, но зато точно знает, где хранится оригинальный загрузчик.

Вот и все! Теперь вы знаете, как устроена очередная "компьютерная зараза" и как ее распознать и обезвредить. Будем надеяться, что эта статья попадет вам в руки до того, как вирус успеет себя проявить. В заключение сформулируем несколько положений, ради провозглашения которых собственно и писалась данная статья.

1. Чтобы написать вирус, много ума не надо.
2. Следствие п.1. Чтобы разобраться с вирусом, тоже много ума не надо.
3. Не так страшен вирус, как его малюют авторы антивирусных программ.
4. Помогай себе сам.
5. Учи аппаратуру и операционную систему, с которой работаешь.
6. Не зазнавайся! Всегда может найтись "технокрыса", знающая систему лучше тебя. ■

Литература

1. **Н.Н.Безруков. Компьютерная вирусология. Справочное руководство. К. УРЕ, 1990.**
2. **Кит Педлер, Джерри Девис. Мутант-59. М.: Правда, 1991.**

Пример 2. Область данных вируса "6-е марта"

```
550F:0000  E9 00AC                    jmp   Vir_Start
550F:0003  F5 00 80 9E                db    0F5h, 00h, 80h, 9Eh
550F:0007  02           Heads_Num     db    2
550F:0008  000E         OrigSectAdr   dw    0Eh

550F:000A  0015 9F00    OldInt13      dd    9F000015h
```

■

Передовые технологии 90-х годов

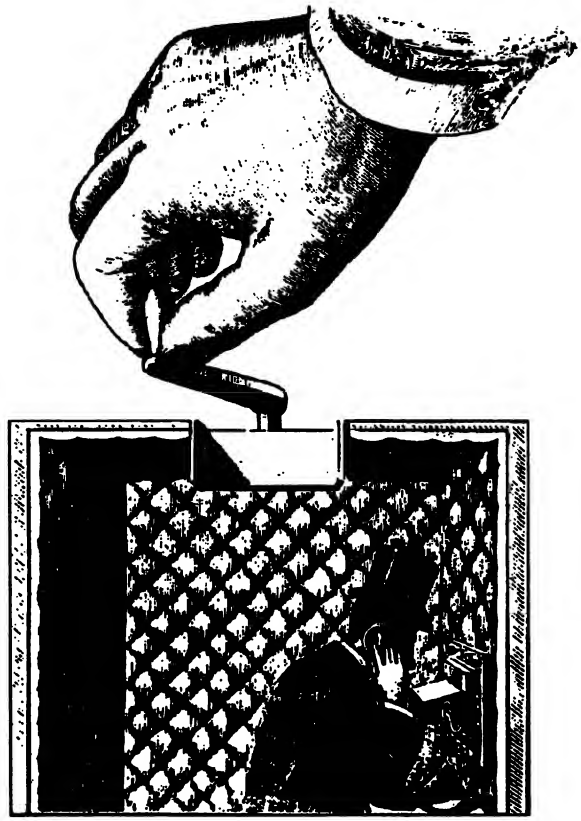
Появление пентопов

Келвин Кларк,торговый агент продукции фармацевтической фирмы Marion Merrel Dow pharmaceutical company,регулярно обзванивал 300 врачей,потребителей продукции компании. Чтобы следить за текущими делами,ему приходилось вести записи в восьми записных книжках. Несколько месяцев назад он заменил эти записные книжки двухкилограммовым портативным компьютером и электронной "ручкой ", связанной с ним. Теперь он может вызывать любые данные, связанные с его клиентами за считанные секунды, просто касаясь электронной ручкой определенных образов или слов на экране.

Согласно прогнозам некоторых аналитиков, эти компьютеры вскоре станут столь же популярны,как и компьютеры типа laptop – лэптоп (в дословном переводе "наколенные", то есть портативные компьтеры,находящиеся внутри корпуса,похожего на дипломат,с которыми можно работать, раскрыв "дипломат" и положив его на колени). По аналогии с термином лэптоп новые компьютеры назвали pentop (pen в переводе с английского означает ручку для письма).

Довольно забавно, что такое древнее средство коммуникации, как обычная ручка для письма, вызвало к жизни новое направление в компьютерной технологии. Компьютеры с электронной ручкой дают возможность пользователям вводить данные и манипулировать ими без применения клавиатуры или мыши. Эти компьютеры портативны и работают на батарейках. Более того,они могут считывать слова и цифры с экрана. То, что они предлагают пользователям – это простота и эффективность обычного персонального компьютера, соединенные в устройстве,которое совсем не похоже на обычный компьютер и не требует умения обращаться с клавиатурой и системой команд.

Этот тип компьютеров является просто развитием того,с чем люди имеют дело каждый день – авторучкой, что делает их особенно привлекательными. Они не отпугивают людей и поэтому должны оказаться особенно успешными. Ожидается,что ежегодная продажа компьютеров этого типа возрастет с 50 000 в 1991 году до 250 000 в 1992 году , а в 1995 увеличится до 2 000 000 штук за год.



Выходите на связь

Часть I

Е. Равич

Для кого написана эта статья

Если вы уже овладели навыками работы на персональном компьютере (ПК), то в какой-то момент перед вами может встать вопрос: какие еще существуют возможности для повышения продуктивности работы в принципе, как без больших потерь времени можно обмениваться информацией с коллегами, чем можно заменить ставшее привычным копирование дискет друг у друга. Вы наверняка уже слышали о сетях передачи данных, но никогда не пользовались их услугами. Возможно, ваши друзья живут в других городах, и вы хотите поделиться с ними имеющимися у вас программными продуктами. Что вам в этих случаях делать? Ответ достаточно прост – воспользоваться средствами связи. Область передачи данных с участием компьютеров практически необъятна, но, к счастью, вам совсем не требуется быть экспертом по коммуникациям, чтобы обмениваться файлами со своего рабочего места. Итак, вы начинаете использовать коммуникационные возможности вашего ПК.

Терминология

Если коммуникации – сравнительно новая для вас область, то вам придется потратить некоторое время, чтобы ознакомиться с используемой терминологией и общепринятыми стандартами. Некоторые термины вам уже знакомы, но мы уточним еще раз их смысл при решении конкретной задачи - связи вашего ПК с другими компьютерами.

Для передачи данных необходимо наличие оборудования передачи данных (Data Communications Equipment – DCE) и оконечного оборудования (Data Terminal Equipment - DTE). В роли DTE может выступать ПК пользователя, терминал, кассовый аппарат и многое другое. Роль DCE заключается в подключении DTE к каналу связи – синхронному или асинхронному. При этом DCE реализует один из интерфейсов, а на более низком уровне – набор протоколов передачи данных. Примером DCE является модем, подключаемый к коммуникационному порту вашего ПК и обеспечивающий выход на телефонную линию. Телефонная линия в свою очередь может быть как обычная (коммутируемая), так и выделенная. При использовании обычных телефонных линий вы соединяетесь с абонентом так же, как при простом телефонном звонке, то есть рискуете услышать шумы и сигнал "занято". Выделенные каналы обеспечивают су-

щественно более высокие скорость и качество связи и отсутствие блокировки, но пока еще очень дороги.

Важным для нас является понятие скорости передачи данных, которая обычно измеряется в битах в секунду (бит/с). Обычные телефонные каналы в СССР не позволяют передавать данные с высокой скоростью (более 4800 бит/с), более того, трудно гарантировать и скорость 2400 бит/с. Опыт показывает, что передача данных чаще всего происходит на скоростях 600 - 2400 бит/с. При этом необходимо учесть, что в реальном сеансе связи кроме данных пользователя передается еще некоторое количество служебной информации, определяемое используемым протоколом.

Протокол передачи данных – это набор соглашений о форматах передаваемых данных, методах синхронизации и защите от ошибок. Поэтому при номинальной скорости передачи 2400 бит/с, непосредственно данные пользователя передаются со скоростью примерно 200 - 220 байт/с. Для передачи данных на различных скоростях разрабатываются специальные рекомендации, в частности, их издает Международный консультативный комитет по телефонии и телеграфии (CCITT). Скорости 1200 бит/с соответствуют рекомендации V.22, а скорости 2400 бит/с – V.22 bis .

Существуют несколько уровней иерархии протоколов. Далее мы рассмотрим уровни, имеющие отношение к нашей задаче.

Примером самого нижнего уровня протоколов (уровень 1 или физический) является хорошо вам известный RS-232-C и его дополнение V.24, определяющие механические, электрические и логические характеристики асинхронного последовательного порта. При асинхронной передаче каждый передаваемый байт обрамляется "стартовым" и "стоповым" сигналами, распознаваемыми принимающим устройством. Это наиболее дешевый метод передачи данных, так как он не требует сложной системы синхронизации между передающим и принимающим устройствами. Работой порта управляет универсальный асинхронный приемник-передатчик (адаптер последовательного порта, UART). Через последовательный порт могут подключаться многие устройства, например, "мышь". Как правило, в ПК установлено несколько (2-4) последовательных порта, что позволяет независимо обращаться к нескольким устройствам. Одно из них может быть и модемом. Синхронную передачу данных мы пока не рассматриваем.

На уровне 2 (канальном) вариантов используемых протоколов существенно больше, и правильный выбор протокола может значительно увеличить скорость передачи данных и резко снизить количество ошибок при передаче. В процессе накопления опыта передачи данных разработчики предлагают все новые и новые протоколы, различающиеся методами коррекции ошибок, размером передаваемых блоков, правилами синхронизации. В меню протоколов современных коммуникационных программ обычно можно сделать выбор между 5-10 встроенными протоколами и несколькими внешними. Примером широко известного протокола, применявшегося ранее при передаче данных между ЭВМ разных типов, является KERMIT. Знакомый всем код ASCII также может использоваться в качестве протокола для передачи текстовых файлов, но без всякой коррекции ошибок.

Наиболее распространены в настоящее время следующие протоколы.

1. XMODEM и его разновидности, различающиеся длиной блока данных и методами контроля ошибок – один из самых распространенных протоколов, относительно медленный и не очень надежный. Первые версии XMODEM были разработаны еще в 1977 г. Версия WXMODEM более эффективна и позволяет существенно повысить скорость передачи данных.

2. YMODEM и его разновидности – более быстрый протокол с варьированием длины передаваемого блока в зависимости от качества линии, достаточно надежный и популярный протокол, использующийся в большинстве коммуникационных программ.

3. ZMODEM – относительно новый, но очень популярный протокол с повышенной надежностью, изменением длины передаваемого блока, многими другими возможностями.

4. Протоколы фирмы Microcom : MNP (Microcom Network Protocol) различных уровней. Скажем только, что MNP 4 обеспечивает коррекцию ошибок, а MNP 5 – еще и сжатие данных, что, как считается, приводит к 200% производительности при передаче данных. Протокол MNP 5 реализуется в некоторых модемах аппаратно. Этим достигается примерно 20-процентное увеличение скорости по сравнению с программной реализацией. Есть, впрочем, данные о плохой сочетаемости ZMODEM и MNP 5 при совместном использовании.

Кроме перечисленных можно упомянуть достаточно сложный и развитый дуплексный протокол BIMODEM, новые протоколы LINX и PUMA, рекламируемые как высокоскоростные и очень надежные, а также протокол Megalink, реализованный в программе GT Powercomm. Протоколы могут быть рассчитаны на передачу как одиночных файлов, так и нескольких файлов в одном сеансе (пакетные протоколы).

Подводя черту, скажем, что вопрос о преимуществах какого-либо протокола может быть решен только после накопления статистики в каждом конкретном случае. Слишком уж различаются условия передачи данных в местах разработки протоколов и местах их применения, то есть у нас. Чем больше размер передаваемого блока, тем выше в принципе скорость передачи, но на зашумленных линиях это срабатывает с точностью до наоборот : приходится повторно передавать большие блоки в случае прихода сигнала об ошибке. Если у Вас размер передаваемого файла не кратен длине блока, то вы вынуждены передать плохо заполненный последний блок (при длине файла 1025 байт и длине блока 1024 байта во втором блоке будет передано 1023 символа конца файла !). Увеличение длины контрольных полей также влечет передачу лишней информации (так называе-

мый overhead), а также потерю времени на собственно контроль при приеме и передаче данных. Вместо использования сжатия данных в протоколе иногда проще упаковать файл заранее, чтобы не увеличивать продолжительность сеанса связи. Сами протоколы (драйверы протоколов) также могут реализовываться с разной эффективностью. Золотую середину здесь найти сложно. Некоторые коллективы советских программистов пошли по пути разработки и реализации собственных протоколов. По опыту работы с коммуникационными программами можно сказать, что при среднем качестве линий связи возможностей протокола ZMODEM обычно бывает достаточно. Не пользуйтесь только вариантами протоколов, рассчитанными на применение высокоскоростных модемов и высококачественных линий связи, например, XMODEM-G и YMODEM-G. Единственное, что вы всегда должны учитывать – выбранный вами протокол должно реализовывать и программное обеспечение, установленное у вашего абонента, поэтому в процессе сеанса связи договоритесь , какой именно протокол вы выбираете для данного сеанса. Постепенно у вас накопится ценный личный опыт и вы поймете, в каких ситуациях у вас бывает меньше всего проблем.

user

Как использовать коммуникационные возможности вашего персонального компьютера?

Модемы : стандарты и разновидности

Какую из сотен предлагаемых моделей модема выбрать ? Точного ответа на этот вопрос, к сожалению, у меня пока нет. Требования к идеальному модему можно предъявить примерно такие : модем должен полностью реализовывать существующие протоколы и соглашения, устойчиво работать на телефонных линиях нашего (как правило, очень низкого) качества, легко подключаться к вашему ПК, аппаратно выполнять коррекцию ошибок, быть совместимым с существующим программным обеспечением и при этом быть доступным по цене. Обычно в каждой конкретной модели все эти требования не выдерживаются.

Рассмотрим требования к модемам подробнее. На имеющихся сейчас в нашей стране телефонных линиях (я не рассматриваю выделенные телефонные каналы, мало доступные большинству пользователей ПК) практически невозможно добиться скорости передачи данных выше, чем 2400 бит/с. Реальная скорость в сеансе может быть и 600 бит/с, поскольку модемы обычно обладают способностью сбрасывать скорость передачи в зависимости от состояния линии связи в конкретном сеансе (этот режим называется Fall back). Для передачи данных на скоростях 1200 и 2400 бит/с, как мы уже знаем, разработаны международные рекомендации V.22 и V.22 bis. Поддержка этих рекомендаций – первое требование к модему .

Далее модем должен быть HAYES-совместимым, то есть реализовывать систему команд, используемую в модемах одного из крупнейших изготовителей модемов – фирмы HAYES. Это позволяет надеяться на совместимость с программным обеспечением и с другими модемами, с которыми вы можете захотеть связаться. Для примера скажу, что выпускаемый в СССР модем Лександ 2400 этому требованию не удовлетворяет. Поэтому работать с таким модемом можно только с помощью прилагаемой к нему программы.

Об устойчивости работы модема заранее сказать невозможно. Некоторые модемы специально тестировались для работы в наших телефонных сетях и показали хорошие результаты (например, модемы OCTOCOMM, US Robotics и SmartLink), но вряд ли вы купите именно их. Поэтому пока мы конкретных рекомендаций дать не можем. Некоторые модели модемов реализуют какую-то форму защиты от помех и коррекцию ошибок аппаратно (в частности, протокол MNP). Стоит аппаратная реализация дороже.

Конструктивно модемы могут быть выполнены в виде платы, которую вам нужно вставить в свободное гнездо на системной плате вашего ПК (internal modem), или в виде отдельного блока (external modem). В случае встраиваемого модема вам необходимо убедиться, что у вас еще есть свободное место на системной плате. Если же модем внешний, то он подключается к свободному последовательному порту ПК, а его источник питания – в обычную розетку. Проверьте сразу, что источник питания рассчитан на используемое у нас сетевое напряжение (220 В, 50Гц). Как правило, к модему прилагается кабель для подключения к телефонной сети, который необходимо переделывать из-за несоответствия телефонных разъемов (в СССР используются двухпроводные телефонные линии, а к модему обычно прилагается четырехпроводной кабель). Если у вас только одна телефонная розетка, подключите модем параллельно вашему телефону.

В большинстве случаев проблем с подключением внешних модемов не бывает. При встраивании же модемов возможны неприятности, в частности, из-за того, что модем конфликтует с другими устройствами, использующими последовательные порты или с адаптерами портов. Вам может понадобиться обращение к документации по устройству системной платы вашего ПК, и возможно снятие лишних перемычек или даже адаптеров портов. После того как вы установили встраиваемый модем, необходимо в соответствии с его инструкцией установить переключатели, которые могут быть как на плоскости платы, так и на ее торце. Дальнейшую настройку вы выполните из коммуникационной программы. Внешние модемы не содержат снаружи переключателей и настраиваются только программно, что

существенно удобнее и быстрее. Автору известны случаи, когда некоторые встраиваемые модемы установить вообще не удавалось. Наибольшее количество неудач происходило с относительно дешевыми и простыми моделями малоизвестных азиатских фирм, документация к которым обычно очень лаконична и может содержать ошибки. Поэтому, видимо, покупать самые дешевые модемы не стоит.

Программы для коммуникаций : основные возможности

Допустим, у вас уже есть уверенность в том, что ваш модем будет устойчиво работать, вы его правильно установили и разобрались в терминологии, относящейся к передаче данных по телефонным каналам. Чего вам недостает ? Правильно, хорошей программы, которая бы достаточно понятно и корректно работала и при этом обеспечивала все ваши высказанные и невысказанные пожелания. Не писать же программу связи самому (хотя возможно и это) ! К счастью, такие программы появились давно, непрерывно совершенствуются и вполне доступны. Обычно одна из таких программ входит в поставку модема. Многие программы предлагаются в режиме FREEWARE или SHAREWARE (как, например, и широко используемые архиваторы) . Некоторые программы вы можете найти в популярном электронном журнале СОФТПАНОРАМА. Если у вас есть такая возможность – купите коммерческую версию с поддержкой фирмы - изготовителя, но в любом случае нужно представлять себе основные возможности таких программ, хотя бы наиболее известных.

Начнем знакомство с многофункциональных программ, обеспечивающих обмен файлами с поддержкой нескольких протоколов. Насколько мне известно, в СССР в настоящее время наиболее распространены программы Procomm Plus фирмы Datastorm, MTE и MTEZ фирмы MagicSoft, TELIX фирмы Exis и BITCOM фирмы BIT Software. Практически все программы для коммуникаций допускают работу в нескольких режимах : терминальном, работы в меню, эмуляции терминала, автоответа и других.

Рассмотрим ваш первый сеанс. Если вы только начинаете работу с программой, вам нужно установить конфигурацию вашей системы : тип модема, к какому порту он подключен, строку инициализации модема, режим набора номера абонента, режим работы терминала, параметры сеанса связи (скорость передачи данных, контроль по четности, количество стоповых битов) и многие другие параметры, вплоть до предпочтительного цвета экрана и отдельных служебных полей. Почти все эти параметры можно установить один раз и запомнить в файле конфигурации. При этом вы работаете в системе вложенных меню. Для большинства параметров обычно можно выбрать значения из предлагаемого списка.

После этого вы можете занести нужные вам телефонные номера в телефонную книгу (dialing directory). Емкость телефонной книги обычно достаточна для большинства применений (200 номеров и более). В дальнейшем вместо набора номера вы просто обращаетесь к телефонной книге и набор делается программным образом. Для каждого телефонного номера вы можете зафиксировать конкретные параметры сеанса связи, а в некоторых программах – связать с ним сценарий будущего сеанса. Кроме этого, если вы собираетесь принимать и передавать файлы, то вам может быть предложено определить директории для приема и передачи. После предварительной настройки вы можете решить для себя, в каком режиме вы хотите работать дальше.

В терминальном режиме вы набираете команды вашего модема вручную, но в соответствии с тем, что именно написано в документации, прилагаемой к модему (обратите на это внимание, так как большинство HAYES-совместимых модемов поддерживает только часть полного набора команд). К форматам команд модема мы еще вернемся.

Большинство пользователей обычно работает в режиме выбора из меню. Это существенно упрощает организацию сеанса и избавляет от необходимости помнить команды модема, хотя и не реализует все его возможности. Меню типичной программы позволяет по нажатию управляющих клавиш переходить из одного режима в другой, получать подсказки, непосредственно устанавливать связь, просматривать директории , выходить в ДОС, вести протокол сеанса связи и так далее, в зависимости от конкретной программы.

В режиме эмуляции терминала вы можете устанавливать связь с удаленной ЭВМ и использовать ее ресурсы (например, работать на ЭВМ типа IBM/370 или VAX) или работать как удаленный пользователь локальной сети – естественно, при соответствующем оборудовании у вашего абонента.

Режим автоответа позволяет вам обрабатывать приходящие звонки в режиме ожидания. Подробно о работе во всех режимах вы можете узнать в документации по вашей программе.

Большинство коммуникационных пакетов имеет в своем составе средства для написания так называемых сценариев (scripts) сеанса связи. Как правило, язык описания сценария предоставляет пользователю десятки и сотни функций, позволяющих полностью автоматизировать типовые сеансы связи и даже написать электронную почту. Сценарии могут компилироваться специальным компилятором сценария, входящим в состав пакета, или интерпретироваться основной программой. Сами языки внешне могут выглядеть похожими на привычные языки программирования, например на язык Си (язык SALT для программы TELIX). Готовые сценарии используются при запуске коммуникационной программы (имя сценария передается в качестве параметра), связываются с конкретным телефонным номером (в телефонной книге), а также могут быть вызваны из меню программы в процессе работы. Конкретная техника работы со сценариями обычно описана в документации, а несколько готовых типовых сценариев, например, для связи с общедоступными (на Западе) электронными почтами и сетями, дается в виде ис-

ходных текстов. Постепенно вы научитесь писать все более и более сложные сценарии.

Другой класс программ – так называемые внешние протоколы, необходимость использования которых возникает, если Ваша основная коммуникационная программа не реализует какой-либо новый и эффективный протокол. Примером таких программ могут служить DSZ фирмы Omen Technology (протокол ZMODEM) и BIMODEM фирмы Eric Labs (протокол BIMODEM). Внешние протоколы могут вызываться из основной программы. Есть также разновидности внешних протоколов, предоставляющие примитивную оболочку для автономного использования.

Несколько слов об электронных почтах, которые уже упоминались выше. Для вас электронная почта (E-Mail), а также доски объявлений (Bulletin Board System, BBS) – это дополнительные возможности по обмену информацией через линии связи. В СССР уже есть целый ряд BBS, доступных обычным пользователям ПК. Телефоны, по которым вам нужно дозваниваться, можно узнать во многих советских компьютерных журналах и выпусках СОФТПАНОРАМЫ. Список телефонов постоянно увеличивается, но по собственному опыту могу сказать, что дозвониться бывает довольно сложно. В каждой системе есть начальное меню, которое появляется после успешного соединения. Дальнейшая работа выглядит как выбор в системе вложенных меню. Недавно стала доступна система электронной почты RELCOM, разработанная кооперативом ДЕМОС. Абонирование в RELCOM'е частных лиц пока стоит не очень дорого. Абонент имеет возможность работать с почтовыми ящиками и пользоваться другими услугами электронной почты, в частности, выходить в некоторые международные сети передачи информации (за дополнительную плату). ■

Зарубежная компьютерная хроника

1991 год, октябрь

IBM и Apple переходят от соперничества к сотрудничеству. Apple будет использовать процессор IBM в будущих поколениях компьютеров Macintosh. Интегральные схемы будет поставлять Motorola. Предполагается также разработка новой операционной системы, совместимой с компьютерами обеих компаний.

Фирма Apple обнародовала финансовые результаты четвертого квартала (объем продаж возрос на 11% по сравнению с тем же кварталом прошлого года) и сообщила о совместном проекте с японской фирмой Sony по созданию портативных лэптоп-компьютеров Macintosh.

---

25 октября Род Кэнион, президент и один из основателей фирмы Compaq Computer, производителя IBM-совместимых ПК, потерял работу. Вместо него назначен Экхард Пфайффер. Увольнение последовало за новостью о первой в истории фирмы квартальной потере.

1991 год, ноябрь

Fujitsu объявила о падении на 41% прибыли за шесть месяцев, предшествовавших 30 сентября (по сравнению с тем же периодом прошлого года). Большая часть потерь обусловлена слабым спросом на 4-мегабитные микросхемы памяти. Ранее в октябре другие японские электронные компании также объявили о падении прибыли и сокращении основных расходов.

Fujitsu оставила планы подарить суперкомпьютер стоимостью в 15 млн. долларов для американского проекта изучения парникового эффекта после давления со стороны американского правительства и корпорации Cray Research, которые расценили этот шаг как демпинг.

Три крупные японские полупроводниковые компании заявили, что они надеются вскоре достигнуть соглашения о совместном проекте по разработке полупроводников для телевидения высокой четкости. NEC, Matsushita Electric и Mitsubishi Electric ожидают, что к ним присоединится по меньшей мере один американский производитель микросхем.

IBM и Intel договорились о разработке компьютера на одной кремниевой микросхеме, что должно существенно снизить стоимость персональных компьютеров.

Крупнейший в мире производитель потребительской электроники Matsushita сообщила, что ее прибыли за предшествующие сентябрю шесть месяцев упали на 29% (по сравнению с прошлым годом) до 197,5 млрд. иен (1,4 млрд. долларов). Причины: покупка американского конгломерата развлечений MCA, слабый спрос на продукцию фирмы и усиление иены.

1991 год, декабрь

9 декабря на встрече с аналитиками Уолл-стрита председатель IBM Джон Акерс вновь изложил свои планы по "перестройке" IBM путем предоставления отдельным подразделениям фирмы большей автономии. Аналитики считают, что в ближайшее время реорганизация не повлияет на результаты деятельности фирмы. Большинство из них сомневается в том, что в 1992 году эффективность работы IBM возрастет. 6 декабря курс акций IBM упал до самой низкой точки за 12 лет (89 долларов). 9 декабря сразу после встречи цена акции упала еше ниже (88 3/4 доллара).

По данным американской Ассоциации полупроводниковой промышленности (Semiconductor Industry Association) отношение "заказы-счета", основной показатель состояния рынка электронных компонент, возросло в ноябре до 1,0 в результате роста числа новых заказов на 5,7%.

Отношение "заказы-счета", равное 1,0, означает, что на каждые 100 долларов проданной продукции производитель получает новых заказов также на 100 долларов. В октябре этот показатель составил 0,96. Он держался ниже 1,0 с июля месяца, отражая спад в американской полупроводниковой промышленности.

Группа The Lazard House объявила о крупнейших сделках 1991 года, в которых она выступала в качестве финансовогосоветника. Среди них приобретение фирмой Borland корпорации Ashton-Tate за 440 млн. долларов, продажа фирмой Borland I млн. акций на сумму 57 млн. долларов и приобретение корпорацией Compaq доли в фирме Silicon Graphics за 135 млн. долларов

стр. 58 ▶

ФИРМА СТИПЛЕР

ЭТО НОВЕЙШИЕ ИЗДАТЕЛЬСКИЕ ТЕХНОЛОГИИ ИЗ ПЕРВЫХ РУК!

МЫ ЗНАЕМ, КАК ЭТО ДЕЛАЕТСЯ:

НАСТОЛЬНАЯ ПОЛИГРАФИЯ
ТЕХНИЧЕСКИЕ ИЛЛЮСТРАЦИИ
ГРАФИЧЕСКИЙ ДИЗАЙН
АВТОМАТИЗАЦИЯ ДЕЛОПРОИЗВОДСТВА

У НАС ВЫ НАЙДЕТЕ ВСЕ НЕОБХОДИМОЕ
ДЛЯ ПРОИЗВОДИТЕЛЬНОЙ РАБОТЫ
КОМПЬЮТЕРА В ВАШЕМ ОФИСЕ.

ПОЛНЫЙ ЦИКЛ ПРОФЕССИОНАЛЬНЫХ ТЕХНОЛОГИЙ ПО ОБЩЕДОСТУПНЫМ ЦЕНАМ.

STEEPLER

МОСКВА 119034, Кропоткинская, 40. Телефон: 246-81-92

Портативные компьютеры

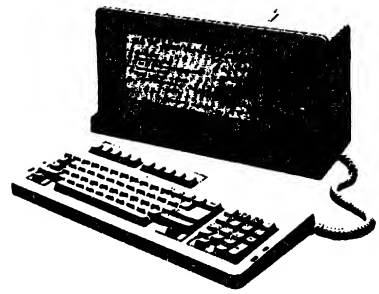


Рис. 2. Sharp PC–7000

Т. Макогон

Менее чем за год технология портативных компьютеров совершила качественный скачок от маломощных, но компактных компьютеров типа TOSHIBA T1000, либо довольно мощных, но увесистых (до 12 кг), к моделям, сочетающим преимущества обоих типов. Тип Notebook (записная книжка) может иметь все типы процессоров: от 8088 до

Рис.1. Tandy 102

80386SX с частотой 20 МГц, математический сопроцессор, до 18 Мбайт памяти, VGA дисплей, винчестер до 80 Мбайт, дисковод на 1,44 Мбайт, порты для подключения самой разнообразной периферии, и все это умещается в корпусе с размерами обычного листа бумаги, толщиной в 4 см и весом 2,5 кг.

Разработка компактной вычислительной техники началась с создания фирмой Texas Instruments в 1969 году первого портативного терминала для ввода данных. Компании стремились выпускать более мощные компьютеры все меньших размеров. По мере развития технологии удавалось делать относительно небольшие машины. И в 1983 году фирмой TANDY был изготовлен первый лэптоп (т. е. наколенный компьютер). Это была модель Tandy 102 (рис.1), имевшая встроенный LCD (жидкокристаллический) экран 40x8 символов, 24K памяти, клавиатуру стандартного размера, Бейсик и другие встроенные программы, встроенный модем и возможность подключения внешнего 3.5" дисковода на 100K, работающего на батарейках или от сети. Корпус имел размеры 32x3x20 см.

Потребность к мощности портативного компьютера росла быстрее возможностей умещать все части в небольшом корпусе. Так, появились модели с 286 процессором, винчестером до 100M и MGA или EGA CRT дисплеем размером 9" по диагонали с весом около 12 кг. Размеры были соответствующими – порядка 42x20x45 см. Их переносимость состояла в прикрепленной к корпусу ручке. Они также могли иметь плоский LCD или GAS PLASMA CGA/EGA экран, что уменьшало размеры до 40x19x24 см и вес до 8-10 кг. Примером может быть модель Sharp PC-7000 (рис.2). Компьютеры типа лэптоп с теми же характеристиками: 286 или 386 процессор, EGA LCD или GAS PLASMA экран, 3.5" дисковод на 1.44 M, аккумулятор для автономной работы на 1-3 часа имели вес около 7 кг и габариты порядка 37x36x9 см. Такие лэптопы выглядят как модель TOSHIBA 5100 (рис.3). Некоторые из этих портативных компьютеров имели 1-2 гнезда для плат расширения, что делало их малоотличимыми по возможностям от настольных (desktop) моделей.

Вместе с тем выпускались более компактные модели типа NEC MultiSpeed, имеющие 640K памяти, два 3.5" дисковода по 720K, CGA дисплей с подсветкой и батарею для работы до 6 часов. Но процессор был 80C88 с частотой 9.54 МГц. За счет сокращения мощности получался выигрыш в весе – 5 кг. Смена курса повышения мощности за счет размеров произошла в 1989 году, когда на рынке появились винчестеры нового формата 2.5" весом всего в 300 граммов. Этот

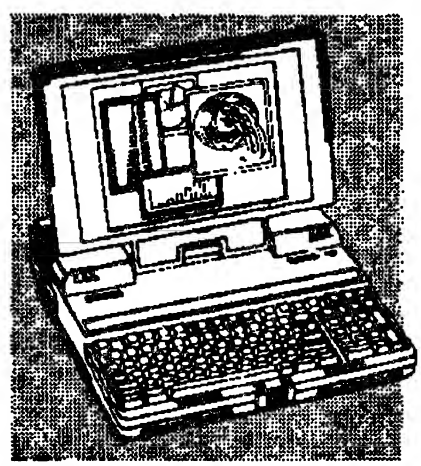


Рис. 3. Toshiba 5100

скачок техники позволил выпускать компьютеры, весящие менее 4 кг. Такой винчестер в компьютере GRiDLITE имел время доступа 28 мс. Одновременное совершенствование наборов микросхем и аккумуляторов также способствовало снижению веса и размеров.

Новое поколение SX Notebook делает существующие портативные компьютеры практически устаревшими. Но заменят ли эти системы также и настольные компьютеры? Хоть они и очень близки к этому, но пока необходимо преодолеть два основных препятствия: относительно небольшие по объему жесткие диски и малую расширяемость. В первом вопросе дело стоит за технологией. Уже выпускаются диски объемом 80M. Примерно через полгода их объем вырастет до 120M, что является приемлемым для интенсивной работы. Порты расширения внутри Notebook есть только для расширения памяти и подключения модема. Для остального в них просто нет места, но решением этой проблемы послужит docking station – станция подключения. Это блок с источником питания и обычно 4 гнездами для 16-битовых плат расширения, одно из которых занято картой соединения с лэптопом. Цена на них меняется от $299 у Tandon NB/386 SX до $1499 у Compaq LTE 386s. Notebook вставляется в нее при работе и получает возможность подсоединиться к локальной сети или другому устройству. Отсутствие цвета – это решенный вопрос, и несколько компаний уже выпускают лэптопы с цветным VGA экраном, но такие машины могут работать только от сети из-за высокого потребления энергии экраном.

Мощный действительно портативный лэптоп незаменим при деловых поездках, когда требуется обработка данных, "не отходя от кассы". Если Вас посылают в командировку с заданием узнать характеристики производства, рассчитать оптимальные параметры и дать профессиональные рекомендации на месте, то кроме знаний о том, как это сделать, еще потребуется и компьютер. Можно возить с собой коробки дискет с нужными программами, как большинство людей и делает, но если с вами лаптоп, весящий лишь 2.5 кг со всеми нужными программами, данными и возможностью иметь свежие данные через модем, то работать становится легко и приятно. Также лэптоп с VGA хорошо иметь на конференции, выступая с докладом.

Плюсы лэптопа перед настольным компьютером начинаются с батареи – на него не влияют скачки напряжения или отключение сети. По мощности они не уступают настольным компьютерам: возможности расширения памяти, скорость процессора, математический сопроцессор, экран, дисковод и винчестер практически те же, но меньших размеров и потребляют меньше энергии. Пока идет работа по созданию мониторов более безопасных по низкочастотному электромагнитному излучению, Notebook имеет самый безвредный LCD экран. Мои глаза еще никогда не уставали от этого экрана, в то время как после долгой работы за новым VGA, не говоря уже о EGA с расстроившейся фокусировкой, сильно устают. У LCD экрана не может расфокусироваться или дрожать изображение.

Типичный 386 Notebook имеет процессор 386SX с частотой 16 или 20 МГц, 1-2M памяти, расширяемые до 4-6M, 20M винчестер и VGA экран. При этом объем его корпуса менее 4500 куб.см (30x25x6) и вес не больше 4 кг. На нашем рынке такой 386SX-20 компьютер стоит от 96 до 126 тысяч рублей +5% в зависимости от винчестера на 20 или 40M. На западном рынке выбор гораздо шире, и если вы заинтересовались таким компьютером, то далее приводится список

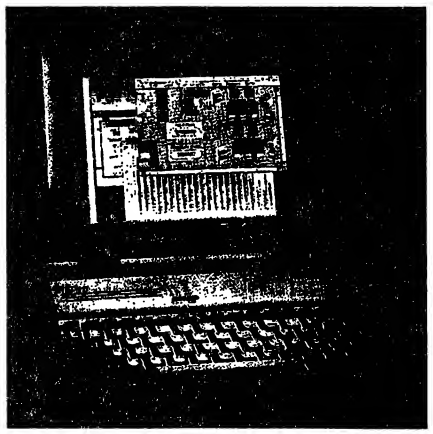


Рис. 4. Compaq LTE 386s

характеристик этих машин (таблица). Цены даны на начало 1991г. По моим оценкам сезонное снижение цен на Западе составляет 5-10%. Все модели имеют дисковод 3.5" на 1.44M и LCD VGA черно/белый экран с передачей цветов оттенками серого. Аккумулятор позволяет автономно работать 2-4 часа, также возможно программируемое отключение питания винчестера и подсветки экрана, что дает увеличение времени работы от аккумулятора. Габариты колеблются от 27.5x20x4.5 см у машин с корпусом типа Texas Instruments до 32x26.7x5.2 см у IBM PS/2 L40SX. Объем винчестера дается в соответствии с ценой. В большинстве машин в среднем за $500 возможно увеличение объема с 20 до 40M и с 30 до 60 или 80M. Кроме того, можно увеличить вдвое размер жесткого диска программой Stacker. Она работает сжимая данные и распаковывая их при обращении к файлу без каких-либо потерь или изменений информации. Ее цена – $149. Каждая модель имеет как минимум 1-2 последовательных и 1 параллельный порты, порт для подсоединения внешнего VGA монитора, могут быть порты для внешнего дисковода, станции соединения, внешней клавиатуры или цифровой клавиатуры, порт специально для мыши и для ручного сканера. При отсутствии таковых можно обойтись и параллельным портом, т.к. выпускаются адаптеры для подключения через него к сканеру, локальной сети Novell, Ethernet, 3Com с ценами от $120 и выше. Если вы привыкли работать с мышью и нельзя без нее обойтись, то ее заменит удобный эквивалент мыши – trackball, сделанный как перевернутая мышь с шаром в ячейке, которым Вы управляете. Trackball легко можно прикрепить сбоку практически к любому Notebook и подключить, как и мышь, через последовательный порт. Два типа trackball продаются компаниями Microsoft и Logitech по цене $175 и $149. Внешний вид у Notebook разных

компаний очень мало отличается, и как пример на рис. 4 показана наиболее мощная из моделей – COMPAQ LTE 386s.

Несмотря на кажущееся однообразие процессоров, все машины будут показывать разную скорость при выполнении той или иной программы. Так, Compaq LTE намного опережает другие машины с 386SX-20 процессором. Достичь этого можно, например, кэшированием памяти, как в картах-ускорителях. Если вам требуется выполнять большие объемы вычислений, то любой из компьютеров имеет гнездо для 387SX сопроцессора. Такие сопроцессоры продаются в модификациях 387SX-16 и 387SX-20 компаниями Intel, IIT и Cyrix по цене от $299 за Intel 387SX-20 до $235 за IIT 387SX-16. Очень небольшое число машин семейства Notebook имеет хорошо скомпонованную клавиатуру. На многих моделях нет клавиш Home, PgUp, PgDn, End, и для их ввода нужно нажимать по две клавиши, а клавиши со стрелками расположены нестандартно: ← → ↕ , а не ← ↕ → из-за малых размеров корпуса.

Только одна из машин – IBM не имеет эмулированных клавиш, т.к. в комплекте поставляется и цифровая клавиатура, прикрепляемая сбоку. В результате получается 101-клавишная клавиатура стандартного размера. Время работы с компьютером в пути можно увеличить вдвое,

user

Сможет ли новое поколение портативных компьютеров заменить настольные компьютеры?

Спецификации 386SX Notebook

Модель	Скорость 386SX, МГц	Память, Мб		Вес, кг	Экран		Винчестер		Цена на начало 1991г.
		Стандарт	Расширен.		Размер по диагонали	Оттенки серого	Объем, Мб	Время доступа, мсек	
ALR Venture	16	1	5	2.8	9"	32	20	21	$2469
AST Premium Executive	20	1	8	3.1	9"	32	20	23	$3582
AUSTIN	20	2	4	3	10"	16	20	24	$2690
BONDWELL B310	16	1	4	3.3	9"	16	80	22	$1848
4COM	16	1	4	3	10"	16	40	25	$2299
COMPAQ LTE 386s	20	2	10	3.4	9"	32	30	21	$4499
COMPUADD Companion	20	2	6	2.5	10"	32	40	22	$3595
DELL 320 N	20	1	5	2.9	9"	16	30	24	$3399
EPSON NB3s	16	1	5	3.1	9"	16	20	22	$2799
EVEREX Tempo lx	16	1	5	2.9	10"	32	20	21	$2199
GOLDSTAR GS520	16	1	4	3.3	9"	16	20	25	$1998
HYUNDAI Super NB	20	2	6	3.1	10"	16	40	24	$2399
IBM PS/2 L40 SX	20	2	18	3.5	10"	64	30	20	$5995
KRIS NSX-1	16	1	5	3.1	9"	16	60	21	$3744
LIBREX 386SX	20	1	4	3.2	9"	16	60	22	$5594
LIGHTSPEED	16	2	4	2.5	9"	16	20	25	$1780
OLIVETTI S20	20	1	6	3.1	10"	32	60	20	$3499
QUEST 386SX	16	1	8	3.1	9"	32	60	20	$4995
SAGER 386SX notebook	16	1	4	3.1	10"	32	20	24	$2775
SAMSUNG NoteMaster	16	1	5	3.1	9"	64	20	23	$3798
TANDON NB 386sx	20	2	16	2.9	9"	16	30	21	$3495
TEXAS INSTRUM Travelmate3	20	2	6	2.5	10"	32	20	20	$3295
TOSHIBA T2000SX	16	1	9	3.1	9"	16	20	19	$4250
TWINHEAD SuperNote	16	2	4	2.7	11"	32	40	25	$2588
ZEOS 386 Notebook	16	1	5	2.9	10"	16	20	21	$2295

имея запасную батарею. В модели TOSHIBA T2000 предусмотрена дополнительная батарея, поддерживающая питание памяти во время смены батарей или в выключенном состоянии, так что выключив машину, можно отдохнуть и затем продолжить работу без перезагрузки и запуска программ. Время перезарядки аккумулятора у Notebook колеблется от 1 до 4 часов, и если вы не на самолете, то можно работать даже с долгосчитающими программами. Поскольку в СССР появилась секторная телефонная связь, то проблема рабочего места исчезает, так как оперативная информация может быть постоянно доступна через модем/факс и секторный радиотелефон.

Улучшение обслуживания клиентов, сокращение бумажной работы и упрощение связи можно получить, дав своим торговым представителям лэптоп для работы. Если вы решили, что Notebook – это именно то, что нужно для работы, то тогда выбирая подходящую модель, останавливайтесь на том, что весит не более 4.5 кг, включая компьютер, модем, батарею и зарядное устройство. В действительности, все эти модели с полным комплектом весят менее 4 кг.

Ищите компьютер с наибольшим возможным экраном и 16 или более оттенков серого цвета. Чем больше оттенков, тем лучше будет смотреться графика. Возможно скоро появятся машины с цветным экраном, работающим от батареи.

Старайтесь найти клавиатуру с готовыми клавишами управления страницами, функциональными клавишами и стандартно расположенными клавишами курсора. Эмулированные клавиши замедляют печать и увеличивают число ошибок.

Для получения всех преимуществ от программ для 386 процессора достаточна частота 16МГц, но для больших расчетов лучше иметь процессор с частотой 20 МГц. Время доступа к данным на винчестере лучше иметь не более 23 мсек.

Большинство программ для 386 процессора требует много памяти. Все машины имеют место как минимум для 4M памяти, что достаточно для одновременной работы 4 программ или работы с multimedia. Также нужен встроенный модем.

Если вы стараетесь найти золотую середину между скоростью, долгой работой от батареи, четким изображением на экране и ценой, то я рекомендую модель Everex Tempo LX. Он не так быстр, как COMPAQ LTE 386SX-20, но довольно дешев по сравнению с остальными, время работы от аккумулятора – 4 часа 13 минут,и зарядить его можно за 1 час 2 минуты. Это незаменимый спутник делового человека при полете Москва – Нью-Йорк Аэрофлотом, когда можно работать во время полета и заряжать батареи во время посадок. Вес в комплекте с адаптером, модемом и батареей – 3.8 кг. С размерами 30.5x25x5 см он легко умещается в кейсе. Большой экран дает самое яркое и четкое изображение. Его клавиатура – одна из самых лучших и удобных. Клавиши находятся на своем месте, и есть специальные клавиши управления страницами, что очень полезно при работе с информацией. Кроме того, клавиатура расположена невысоко от стола по сравнению с другими компьютерами – 2.7 см, и руки при работе не устают. Если вы выберете Everex Tempo LX, то получите хорошую скорость для машины с процессором 386 SX-16, удобный в работе экран и исключительно быстрое время перезарядки.

Современные компьютеры имеют все больше мощности на килограмм веса. Одними из самых мощных в персональных компьютерах считаются RISC процессоры, и несколько компаний уже выпускают лэптопы с ними, но размеры и вес этих машин оставляют желать лучшего. Примерно через год-полтора развитие технологии даст возможность поместить несколько мегабайт памяти, большой винчестер, дисковод на 2.88M и цветной экран в корпус с размерами не больше пачки бумаги. Что касается батарей, то компьютер TOSHIBA T2000SX уже имеет последнее достижение в технологии аккумуляторов – никель-гидридный, запасающий на 22% энергии больше, чем никель-кадмиевый, и не страдающий от эффекта памяти. Создается и новый стандарт портативных компьютеров – penbook. Он будет состоять собственно из экрана и иметь ручку для ввода текста и графиков на экран, 386SX процессор, жесткий диск либо его аналог на микросхемах с питанием от батареи, весить порядка 2 кг и иметь толщину около 2 см при размерах листа бумаги. Фирма GRiD уже выпустила такой компьютер. Однако специалисты считают, что широкого применения penbook не найдет, т.к. скорость ввода рукописного текста меньше скорости при печати на клавиатуре. Дальше компьютер будет становиться все более персональным, получит управление от голоса и уменьшится до необходимых размеров экрана.

В случае, если Вы решили приобрести Notebook, далее приводятся координаты компаний. Если этой информации не хватит, обращайтесь с факсом 135-88-76, код Москвы 095, либо звоните вечером по телефону 137-36-66. ■

Где купить

Notebook:

COMPAQ LTE 386s – Compaq Computer Corp.
P.O.Box 692000
Houston, TX 77269-2000 USA
телефон: 1-800-231-0900

Everex Tempo LX – Everex Systems, Inc.
48431 Milmont Dr.
Fremont, CA 94538 USA
телефон: 1-800-821-0206

Интерфейс сканера:

CAT scanner laptop link – 1-800-2881611

Trackball:

Microsoft BallPoint Mouse – Microsoft Corp.
One Microsoft Way
Redmond, WA 98052-6399 USA
телефон: 1-800-426-9400, 1-206-882-8080

Logitech TrackMan Portable – 6505 Kaiser Dr.
Fremont, CA 94555 USA
телефон: 1-800-231-7717, 1-415-795-8500

При звонках за рубеж учитывайте разницу во времени.

Этюд 2. Рекурсия или не стреляйте из пушки по воробьям

Д. Бройтман

Латинское слово recursio в переводе на русский язык означает "возвращение".

В математике функция называется рекурсивной, если ее значение для данного аргумента определяется через значения той же функции для предшествующих аргументов.

В программировании мы называем функцию рекурсивной, если последовательность операторов, составляющая тело функции, включает в себя один или несколько вызовов самой этой функции.

Рекурсия – очень мощное средство программирования. Поэтому при использовании рекурсии следует соблюдать осторожность.

Существует класс "типично рекурсивных" задач, в которых рекурсия приводит к ясным, изящным и эффективным алгоритмам решения.

Но часто возникает соблазн использовать рекурсивный алгоритм из-за его кажущейся простоты и краткости в тех случаях, когда достаточно на порядок более простой конструкции – повторения (цикла).

В лучшем случае такое употребление рекурсии – просто "дурной тон", о котором Э.Дейкстра отзывается так: "Несмотря на формальную правильность, это мне неприятно, потому что я не люблю, когда из пушки стреляют по воробьям, вне зависимости от того, насколько эффективно пушка справляется с такой работой".

Однако вот пример катастрофических последствий, к которым приводит слепое следование за рекурсивной формулировкой задачи.

Пример знаменитый – вычисление последовательности чисел Фибоначчи:

```
f(0) = f(1) = 1;
f(n) = f(n-1) + f(n-2)
для n >= 2.
```

Механический переход от определения к алгоритму приводит к следующей рекурсивной функции:

```
long int fib1(int n)
{
    if((n == 0)||(n == 1))
        return(1);
    else
        return(fib1(n-1) + fib1(n-2));
}
```

Альтернатива – эффективное вычисление последовательности чисел Фибоначчи с помощью оператора цикла:

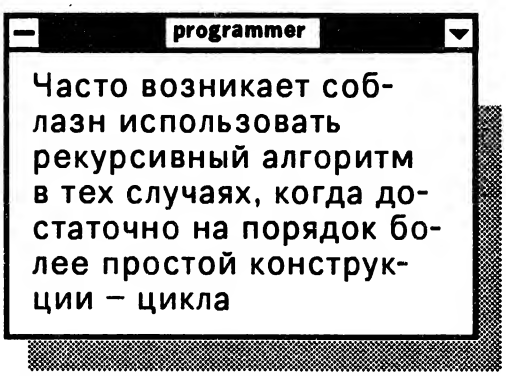
```
long int fib2( int n )
{
    long int a, b, f;

    if(n < 2)
        return(1);
    else {
        a = b = 1;
        while(—n) {
            f = a + b;
            a = b;
            b = f;
            }
        return(f);
        }
}
```

Читателю предлагается оценить время, которое понадобится каждой из этих функций для вычисления, например, 50-го числа Фибоначчи. Если же вы считаете, что программирование – наука экспериментальная, то оттранслируйте их (добавив вызов из функции main) и запустите при n = 50.

Даже если использование рекурсии оправдано, неаккуратное обращение с ней может привести к трудно обнаруживаемым ошибкам, когда программа работает правильно при одних входных данных и неправильно – при других, или, что еще хуже, обычно работает правильно, но для некоторых входных данных порождает бесконечную последовательность вызовов и никогда не завершается.

В этом этюде мы решим несколько классических рекурсивных задач и на их примере познакомимся с алгоритмами, в которых рекурсия не только оправдана, но и необходима. Кроме того, мы научимся строго доказывать завершимость работы и правильность рекурсивных программ.



Ханойские башни

Легенда. В одном из храмов Юго-Восточной Азии находятся три вертикальных стержня, на которые нанизаны 64 золотых кольца разного диаметра. Некогда бог Вишну поместил все 64 кольца на первый стержень так, что диаметр колец снизу вверх уменьшался, и повелел жрецам переместить башню из колец с первого стержня на третий, соблюдая следующее правило: на каждом шаге можно перенести самое верхнее кольцо с одного из стержней на верх другого стержня при условии, что на каждом из стержней кольца будут сохранять форму башни (т.е. их диаметр снизу вверх уменьшается). С тех пор много тысяч лет жрецы днем и ночью перекладывают кольца. Легенда гласит, что когда все кольца окажутся на третьем стержне, наступит конец света.

Напишите программу, решающую задачу по перемещению ханойской башни.

Для того чтобы убедиться в правильности программы, перемещающей ханойскую башню (пример), нам следует доказать три утверждения.

1. Программа завершит свою работу при любом начальном числе колец n = 1 на первом стержне.

2. После завершения работы программы кольца образуют правильную башню на третьем стержне.

3. В ходе работы программы не нарушается правило расположения колец: для любой пары колец на любом из стержней диаметр верхнего кольца меньше, чем диаметр нижнего.

Проведем доказательство индукцией по числу колец n.

При n = 1 работа функции hanoi сводится к переносу единственного кольца с первого на третий стержень. Справедливость утверждений (1 - 3) для этого случая очевидна.

Предположим, что (1 - 3) справедливы при n=k, где k>=1. Тогда для n = k + 1>1 выполнение функции hanoi(k+1, 1, 2, 3) сведется к выполнению следующей последовательности операторов:

```
1. hanoi(k, 1, 3, 2)
```

Согласно предположению индукции этот оператор завершит-

Программа перемещения ханойской башни

```
#include <stdio.h>
#include <dos.h>        /* sleep() */

#define MAX_NRINGS 64       /* Максимальное */
                            /* число колец */

int st[4][MAX_NRINGS];   /* 1,2,3 – стержни */
int nr[4];               /* Число колец на стержнях */
int nmoves;              /* Число перемещений */

/*-------------------------------------------------*/
/*Печать текущего расположения колец на стержнях*/
/*-------------------------------------------------*/
void print_st( void )
{
    int i, j;

    for( i = 1 ; i <= 3 ; i++ ) {
        printf("\n| ");
        for( j = 0 ; j < nr[i] ; j++ )
            printf("%d ", st[i][j] );
        }
    printf("\n");
}

/*-------------------------------------------------*/
/*Установка начального положения колец*/
/*-------------------------------------------------*/
void init( int nrings )
{
    for( nr[1] = 0 ; nrings > 0 ; nr[1]++,nrings-- )
        st[1][nr[1]] = nrings;
    /* Нанизали кольца на 1-й стержень */
    nr[2] = nr[3] = 0;
    /* Два других стержня пусты */
    nmoves = 0;
    print_st();
}

/*-------------------------------------------------*/
/*Функция переносит одно кольцо*/
/*  со стержня n1 на стержень n2  */
/*-------------------------------------------------*/
void move1( int n1, int n2 )
{
    st[n2][nr[n2]++] = st[n1][--nr[n1]];
    sleep(1);          /* Пауза в 1 секунду */
    print_st();        /* Печать текущей позиции */
    nmoves++;
}

/*-------------------------------------------------*/
/*Функция hanoi перемещает верхние nrings колец*/
/*со стержня i1 на стержень i3, используя стержень*/
/*i2 в качестве промежуточного. 1 <= i1,i2,i3 <= 3.*/
/*-------------------------------------------------*/
void hanoi( int nrings, int i1, int i2, int i3 )
{
    if(nrings == 1)
        move1(i1, i3);
    else {
        hanoi(nrings-1, i1, i3, i2);
        move1(i1, i3);
        hanoi(nrings-1, i2, i1, i3);
    }
}

main()
{
    int nrings;

    printf("Число колец: ");
    scanf("%d", &nrings);
    init(nrings);
    hanoi(nrings, 1, 2, 3);
    printf("Перенос колец завершен.\n");
    printf("Число перемещений - %d.\n", nmoves);
    return(0);
}
```

ся, причем в ходе его выполнения условие (3) нарушено не будет, после завершения работы верхние k колец образуют башню на 2-м стержне, а самое большое (k+1)-е кольцо останется на 1-м стержне.

2. move1(1, 3)

Самое большое кольцо переносится с 1-го на 3-й стержень. Очевидно, что при этом условие (3) соблюдается (т.к. 3-й стержень был пуст).

3. hanoi(k, 2, 1, 3)

Башня из k колец переносится со 2-го на 3-й стержень, используя 1-й в качестве промежуточного. Согласно предположению индукции этот оператор также завершит свою работу, условие (3) в ходе работы нарушаться не будет, а после окончания работы k колец образуют на 3-м стержне правильную башню над уже лежащим там самым большим (k+1)-м кольцом.

Утверждения (1 - 3) для n = k + 1 доказаны.

Таким образом, мы доказали, что наша рекурсивная функция перемещения ханойской башни работает правильно для любого числа колец в башне.

Примечание. Для программы, текст которой приведен выше, число колец не должно превосходить MAX_NRINGS.

Функция Аккермана

```
int a( int x1, int x2 )
{
if(x1 == 0)
    return( x2+1 );
else if(x2 == 0)
    return( a(x1-1, 1) );
else
    return( a(x1-1, a(x1, x2-1)) );
}
```

Докажите, что для любой пары неотрицательных целых чисел x1 и x2 вычисление функции Аккермана завершится.

Будем считать, что если в операторе есть несколько обращений к функции (как в последней строке функции Аккермана), то первым вычисляется самое внутреннее обращение.

Множество аргументов A функции Аккермана – множество пар неотрицательных целых чисел (x1, x2), где x1 >= 0, x2 >= 0.

Введем на этом множестве отношение порядка.

(x1, x2) < (y1, y2), если x1 < y1
или (x1 = y1 и x2 < y2).

(x1, x2) = (y1, y2), если x1 = y1 и x2 = y2.

Для любых двух пар из множества A
(x1, x2)<(y1, y2) или (x1, x2) > (y1, y2)
или (x1, x2) = (y1, y2).

В множестве A существует наименьший элемент – пара (0, 0).

Будем доказывать завершимость функции Аккермана индукцией по множеству аргументов A.

Для минимального элемента из A вычисление функции завершается и a(0, 0) = 1.

Предположим, что вычисление функции завершается для любой пары (x1, x2) из множества A такой, что (x1, x2) < (y1, y2), и покажем, что тогда для пары (y1, y2) вычисление функции Аккермана также завершится.

Если y1 = 0, то вычисление функции завершится сразу: a(y1, y2) = y2 + 1.

Если y1 > 0 и y2 = 0, то a(y1, y2) = a(y1 - 1, 1). Так как (y1-1, 1)<(y1, y2), то вычисление по предположению индукции завершится.

Если y1 > 0 и y2 > 0, то a(y1, y2) = a(y1-1, a(y1, y2-1)).

Вычисление внутреннего обращения к функции завершится, т.к. (y1, y2-1) < (y1, y2). Вычисление внешнего обращения также завершится,
т.к. (y1-1, z)<(y1, y2) при любом z.

Таким образом, вычисление a(y1, y2) обязательно завершится.

Использованный нами метод доказательства индукцией по структуре множества аргументов функции называется методом структурной индукции. Это основной метод доказательства правильности рекурсивных программ. Его применение корректно только в том случае, если введенное нами отношение порядка превращает множество аргументов в так называемое вполне упорядоченное множество. Математически строгое обоснование метода можно найти в книге [1].

Задачи для самостоятельного решения

1. Мир состоит из множеств. Каждое множество может быть пустым ({ }), содержать целые числа (атомы) ({1 2 3}), содержать как атомы, так и другие множества ({1 2 {4 5} 8 {9 {} 12 {1 3} 6} }). Повторяться элементы в множестве не могут, хотя в состав разных элементов-множеств могут входить одни и те же атомы.

Заданы:

1) тип "элемент множества":
typedef ... elem;

2) функции работы с элементами и множествами:

- char nil(elem s) – возвращает 1, если s – пустое множество, иначе возвращает 0;
- char atom(elem s) – возвращает 1, если s – атом, если s – множество, то возвращает 0;
- elem car(elem s) – возвращает первый элемент множества s;
- elem cdr(elem s) – возвращает множество s без первого элемента.

Используя функции nil, atom, car и cdr, напишите программу, которая строит из исходного множества новое, состоящее из всех различных атомов, содержащихся на всех уровнях исходного множества. Докажите правильность вашей программы.

Например, из множества
{1 2 {3 4} {1 {5 7} 3} 9}
будет построено множество
{1 2 3 4 5 7 9}.

Подсказка. При доказательстве используйте индукцию по количеству символов (фигурных скобок и атомов) в записи исходного множества.

2. Докажите, что в задаче о ханойских башнях число перемещений колец равно $2^n - 1$ (n - число колец).

3. Попробуйте придумать нерекурсивные алгоритмы решения рассмотренных в этюде задач. ■

Литература

1. Р.Андерсон. Доказательство правильности программ. М.: Мир, 1984.

С целью предоставления владельцам персональных компьютеров информации о фирмах, действующих на компьютерном рынке, и предлагаемых ими продуктах и услугах, "Монитор" начинает публиковать на страницах журнала каталоги фирм и программ.

В каталоге фирм помещается следующая информация о фирме.

- Название фирмы, адрес, телефон, факс.
- Краткое описание продуктов и услуг, предлагаемых фирмой (не более 30 слов).

В каталоге программ помещается следующая информация о программе.

- Наименование программы.
- Как связаться с поставщиком (адрес, телефон, факс).
- Краткое описание программы и ее функций (не более 30 слов).
- Аппаратура и операционная система, необходимые для работы программы.
- Языки, на которых доступны программа и документация.
- Число пользователей в России и в других странах.
- Цена (или диапазон цен).

"Монитор" публикует также объявления фирм, желающих найти высококвалифицированных специалистов по компьютерам и программированию, и объявления частных лиц, ищущих работу в этой области.

Для получения более подробной информации Вы можете позвонить по телефону (095)237-21-36 либо выслать открытку со своими координатами по адресу 117419, Москва, а/я 765.

Малое предприятие НВЦ ПРИНТ. 125057, г. Москва, а/я 71, тел. 155-86-82, 155-45-74.

Предлагаем за рубли электронные ключи защиты программ от несанкционированного использования, изготовленные фирмой "Software Security" (США).

Совместное Советско–Кипрское предприятие Мультибайт. 103001, г. Москва, Вспольный пер., д. 21, тел. 299-68-26, факс 200-17-54.

Программно-аппаратные комплексы локальных вычислительных сетей; базы данных; брокерская контора №9 на Международной Бирже информации и телекоммуникаций.

ASKER, Acquisition & Structuring Knowledge Editor (Среда для Извлечения и Структуризации Знаний)

101477, Москва, ГСП-4, Ермоловой 19, ИППИ АН, лаб. 3, Андрей Свириденко, (095) 138 8781, факс (095) 292 6511 for IntelliSoft, box 10200.

ASKER - интеллектуальная система для создания и уточнения классификаций, рассчитанная на инженеров знаний. ASKER состоит из редактора нейронной сети и синтезатора текстов, доступных из единой среды в стиле Turbo. Система позволяет уточнять знания на реальных данных. Документация и интерфейс созданы в русском и английском вариантах. Необходим IBM-совместимый компьютер, 640K, жесткий диск.

Имеется опыт практического использования ASKER 1.0 в медицине (нефрология, терапия) и экологии.

За рубли система продается впервые. Всего 9 950 рублей за копию

ВНИМАНИЕ !!!
Руководители и специалисты предприятий и учреждений всех форм собственности!

Предприятие "ПАРАЛЛЕЛЬНЫЕ КОМПЬЮТЕРНЫЕ СИСТЕМЫ о.о.о." предлагает оснастить Ваши IBM-совместимые компьютеры следующими программными продуктами:

1. ПЕРСОНАЛЬНАЯ ИНФОРМАЦИОННАЯ СИСТЕМА в составе:
   система разработки, хранения и поиска документов;
   ежедневник, записная книжка, калькулятор;
   календарь рабочего времени;
   система информирования о контрольных датах,
   система перевода валют и разных систем измерения,
   система модемной связи с пользователями такой же системы.
2. СИСТЕМА КОНТРОЛЯ ИСПОЛНЕНИЯ ПОРУЧЕНИЙ, ПРИКАЗОВ, ПЛАНОВ.
3. ПРОГРАММНЫЙ КОМПЛЕКС УПРАВЛЕНИЯ БУХГАЛТЕРИЕЙ в составе следующих подсистем:
   учет основных средств;
   учет финансово-банковских операций;
   учет товарно-материальных ценностей;
   расчет заработной платы.
4. СИСТЕМА "ЭЛЕКТРОННАЯ БИРЖА", обеспечивающая прием и обработку коммерческой информации, поступающей по телефонным линиям связи.
5. СИСТЕМА АВТОМАТИЗИРОВАННОГО СКЛАДСКОГО УЧЕТА.
6. СИСТЕМА УПРАВЛЕНИЯ ОСНОВНЫМ ПРОИЗВОДСТВОМ на промышленном предприятии в составе:
   АРМ конструктора серийного конструкторского отдела; АРМ технолога бюро технической подготовки производства; АРМ нормировщика отдела труда и зарплаты; АРМ диспетчера планово-диспетчерского отдела; АРМ отдела внешней кооперации; АРМ работника планово-распределительного бюро цеха; АРМ экономиста цеха; АРМ склада деталей.

МЫ ОБЕСПЕЧИВАЕМ ОБУЧЕНИЕ ПОЛЬЗОВАТЕЛЕЙ И СОПРОВОЖДЕНИЕ НАШИХ ПРОГРАММ В ТЕЧЕНИЕ 3 ЛЕТ.

Наш адрес: 107082, г.Москва, а/я 2. Тел./факс (095) 269-19-66.

Для более точного выявления интересов наших читателей отдел маркетинговых исследований журнала "Монитор" проводит анкетирование. Результаты анкетирования отразятся на содержании ближайших номеров журнала. Среди читателей, приславших заполненные анкеты, будет проведена лотерея, в которой будут разыграны 2 коробки дискет HD (3" и 5") и 10 бесплатных подписок на "Монитор-92" и "Монитор-Диск-92".

Анкеты высылайте, пожалуйста, по адресу:
117419, Москва, а/я 765.

1. Какие компьютерные журналы Вы читаете?

_____

_____

2. Почему? _____

_____

3. Для чего Вы читаете компьютерные журналы:

☐ чтобы быть в курсе мировых тенденций развития
☐ для знакомства с передовыми западными разработками
☐ для знакомства с последними отечественными достижениями
☐ для решения конкретных задач на работе (подходы, методы, способы, алгоритмы)
☐ чтобы найти описания конкретных программных продуктов
☐ чтобы быть в курсе коньюнктуры рынка
☐ чтобы приятно провести время
☐ затрудняюсь ответить

4. Как, по Вашему мнению, должна быть представлена информация (дайте, пожалуйста, не более трех ответов):

☐ в виде фактов и цифр
☐ в виде переводных западных статей
☐ как обзоры западной прессы
☐ как сравнение отечественных и западных программных продуктов
☐ в виде оригинальных статей отечественных авторов
☐ в виде алгоритмов
☐ в виде текстов программ
☐ в виде рекламных объявлений
☐ затрудняюсь ответить

5. Какой объем должны, по Вашему мнению, занимать в современном компьютерном журнале тексты программ:

☐ текстов программ быть не должно
☐ тексты программ должны занимать _____% от объема журнала

6. Какой объем должны, по Вашему мнению, занимать в современном компьютерном журнале обзоры западной прессы:

☐ обзоров западной прессы быть не должно
☐ обзоры западной прессы должны занимать _____% от объема журнала

7. Какой объем должны, по Вашему мнению, занимать в современном компьютерном журнале статьи по компьютерному бизнесу?

☐ статей по компьютерному бизнесу быть не должно
☐ статьи по компьютерному бизнесу должны занимать _____% от объема журнала

8. Какой объем должна, по Вашему мнению, занимать в современном компьютерном журнале реклама:

☐ рекламы быть не должно
☐ реклама должна занимать _____% от объема журнала

9. Должен ли, по Вашему мнению, журнал специализироваться на конкретной проблематике или ориентироваться на широкие круги читателей:

☐ журнал должен быть специализированным
☐ журнал должен ориентироваться на широкие круги читателей

10. Если вы считаете, что новый компьютерный журнал должен быть специализированным, то на кого, по Вашему мнению, он должен ориентироваться:

☐ на профессиональных прикладных программистов
☐ на профессиональных системных программистов
☐ на профессиональных пользователей
☐ на людей, занимающихся компьютерным бизнесом

11. Какие из перечисленных тем представляют для Вас наибольший интерес (отметьте не более трех пунктов):

☐ объектно-ориентированное программирование
☐ операционные системы и оболочки (DOS, Windows)
☐ СУБД (dBase, Clipper, Paradox)
☐ искусственный интеллект (экспертные системы, нейронные сети, общение на естественном языке, компьютерное зрение)
☐ сети, модемы, телекоммуникации
☐ язык программирования C
☐ язык программирования Pascal
☐ компьютерная графика
☐ САПР
☐ работа на "низком" уровне (прерывания, работа с внешними устройствами)
☐ компьютерные вирусы
☐ компьютерный бизнес
☐ вопросы охраны авторских прав и интеллектуальной собственности
☐ статьи развлекательного характера
☐ рецензии и рефераты книг
☐ информация о ведущих фирмах, действующих на компьютерном рынке
☐ другое

_____

12. Каким образом Вы чаще всего приобретаете компьютерные журналы:

☐ покупаете в книжном магазине
☐ покупаете у торговцев на улице
☐ покупаете на выставках
☐ берете у знакомых
☐ берете в библиотеке
☐ получаете по подписке

13. Сколько, по Вашему мнению, должен стоить новый компьютерный журнал в том виде, каким Вы хотите его видеть:

☐ 2 рубля
☐ 5 рублей
☐ 10 рублей
☐ 15 рублей
☐ 20 рублей
☐ 30 рублей
☐ 50 рублей
☐ более 50

14. В каком городе Вы живете? _____

15. Сколько Вам лет? _____

16. Ваша профессия _____

17. Адрес для корреспонденции или любые координаты для высылки приза _____

_____

_____

Homo computerus

А. Халамайзер

До самого последнего времени мало кто обращал внимание на то обстоятельство, что общение с ЭВМ приводит к изменению самих людей, их поступков, привычек, характеров и даже судеб.

На наших глазах рождается новое социальное явление – homo computerus, что в переводе с латинского означает "человек компьютеризированный".

Остановимся на некоторых, до сих пор не всеми осознанных, особенностях компьютерных профессионалов.

Как разделить сокровище?

Знаете ли вы способ самого справедливого деления буханки хлеба, кучки драгоценностей или чего-нибудь еще на двоих? Механизм простой и изящный: один участник делит на две любые доли (пусть неравные, ему же хуже!), а второй выбирает ту из получившихся долей, которая ему кажется лучше (больше) другой.

Но как быть, если участников трое? И из них любые двое могут быть в тайном сговоре против третьего? Ответ таков: первый делит на три любые доли. Второй выбирает самую по его мнению плохую. Если с этим согласен третий, то ее отдают тому, кто делил, а если не согласен - то ее забирает третий. Затем оставшиеся две доли опять соединяются и уже известным способом делятся на тех двоих, которые пока еще ничего не получили.

Решение (и даже простая проверка решения) таких задач представляется нам чрезвычайно увлекательным. Этот факт не имел бы такого серьезного значения, если бы не выяснилось, что те, кого объединяет с нами любовь к решению и разбору подобных задач, чаще всего оказываются хорошими программистами или математиками.

"Если лицам разных профессий поручить одинаковую работу, то математик сделает ее лучше", - справедливо заметил однажды известный польский матаматик Гуго Штейнгауз.

И наоборот, люди, далекие от ЭВМ и от математики, чаще всего не проявляют к подобным задачам интереса, за исключением, пожалуй, профессиональных юристов.

Интересно ли вам играть в новую игру? Быстро ли вы приспосабливаетесь к новым, непривычным для вас правилам игры? Нравится ли вам получать неожиданные результаты, пользуясь средствами, которые на первый взгляд для этого совсем не подходят? Математики и программисты на эти вопросы обычно отвечают положительно.

Наверное, это не случайно.

Итак, психология, логика мышления и даже поведение профессиональных программистов во многих случаях отличаются от психологии, мышления и поведения "обычных" людей. Этот факт, находящий множество подтверждений, можно объяснить двояко: либо программистами становятся исключительно люди, изначально обладающие некоторыми особенностями, либо на их мышление накладывает отпе-

чаток длительная профессиональная деятельность.

Ответ точный или ответ полезный?

Два человека летели на воздушном шаре, попали в бурю и заблудились. Через несколько часов они приземлились в совершенно незнакомом населенном пункте и, не вылезая из корзины шара, спросили прохожего:

– Скажите пожалуйста, где мы находимся?

– Вы находитесь в корзине воздушного шара, - ответил абориген.

– Наверное, это программист, - задумчиво сказал один из воздухоплавателей другому. – Только программист может дать такой абсолютно точный и абсолютно бесполезный ответ.

Когда специалисту - консультанту в какой-либо технической области задают вопрос по специальности, то хороший консультант задумывается:"А действительно ли спрашивающего интересует именно то, о чем он спросил?" Очень часто бывает, что нужен ответ на другой, несколько более общий вопрос, может быть совет совсем из другой области. Точный ответ на вопрос:"Как лучше всего колоть орехи имеющимся у меня микроскопом?" – гласил бы, что микроскоп удобнее всего держать за тубус, а орех положить на твердую, но не скользкую поверхность. Однако полезный ответ на этот вопрос состоит в том, что колоть прибором орехи не следует, для этого есть специальные инструменты.

Итак, верный, точный, правильный ответ на заданный вопрос часто оказывается бесполезным или даже вредным.

К сожалению, длительное общение с компьютерами, которые сами, как правило, дают только точные ответы, приводит к тому, что профессионалы часто почти автоматически начинают общаться с людьми в такой же строго-бесполезной манере. Когда профессионалу задает вопрос тоже профессионал, то он рассчитывает на формальный, строгий, точный ответ. То есть он старается сформулировать именно тот вопрос, точный ответ на который его на самом деле интересует.

А вот вопрос задает "обычный" человек:

- Могу ли я доехать на этом трамвае до вокзала?

- Да, - отвечает программист.

Ответ верный. Может доехать.

Но трамвай-то идет в данный момент в другую сторону! Спрашивающий может проехать 17 остановок до конца маршрута, затем еще 26 остановок в обратную сторону, и действительно доедет до вокзала. Полезный же ответ состоял даже не в том, что надо проехать на трамвае с тем же номером 10 остановок в обратную сторону, а в том, чтобы воспользоваться останавливающимся тут же автобусом-экспрессом, который будет у вокзала через пять минут и подойдет ближе к перрону.

Однако это ответ на вопрос "Как лучше добраться до вокзала?", а такого вопроса программисту не задавали!

Зачем нужны "профессиональные идиоты"?

"Профессиональный идиот" – это не всегда буквальная характеристика. В коллективах, создающих сложные – и не только компьютерные – системы, так называют высококвалифицированных профессионалов, в чьи задачи входит нахождение слабых мест в сдаваемых разработках. Они получают в свое распоряжение систему непосредственно перед или сразу после прохождения ОТК и стараются вывести ее из строя законными способами.

Когда представитель этой профессии возьмется за очередную модель телевизора, то он, вероятно, попробует:

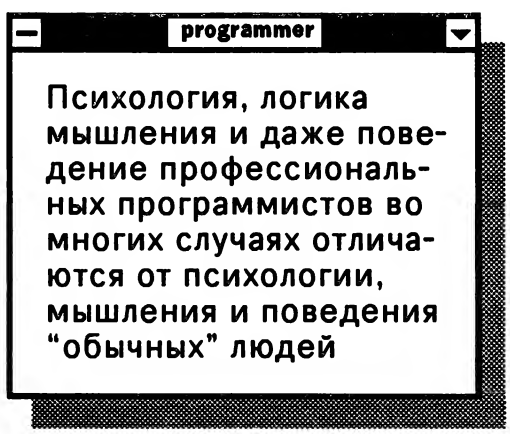
- включить ее в сеть не с тем напряжением - не должно быть ничего хуже сгоревшего предохранителя;
- выключить из сети не выключателем, а просто вынуть вилку из розетки;
- одновременно нажать две или более кнопок выбора программы;
- перевести все регуляторы в крайнее положение (опробовать несколько комбинаций) и т.д.

Работа профессионального идиота чрезвычайно важна. Именно она позволяет находить в системах места, имеющие недостаточную "защиту от дурака".

"Защита от дурака", как и выражение "профессиональный идиот", не метафора, а вполне серьезный, знакомый специалистам термин, относящийся к надежности человеко-машинных систем. Это, иначе говоря, защита от непродуманных действий при пользовании системой - скажем, тем же телевизором или нагревательным прибором.

При неполной проверке особых ситуаций возможно, например, что из-за ошибки одного билетного кассира будет надолго парализована продажа железнодорожных билетов во всем городе или даже в нескольких городах. Поэтому в качестве профессиональных идиотов обычно используются специалисты с большим собственным опытом работы по созданию сложных систем. Они могут чутьем уловить места, где разработчик не предусмотрел достаточной защиты программы от маловероятной ошибки человека или оборудования.

Если кофеварку или другой электрический водонагрева-



тельный прибор включить без воды, прибор выйдет из строя. Поэтому во многих таких приборах предусмотрена "защита от забывчивых": если вода не налита, нагревательный элемент не включается.

Длительная работа в качестве профессионального идиота создает привычку к анализу самых невероятных для "обычных людей" ситуаций. Это может показаться страшно скучным, а сам такой специалист - неприятным собеседником или попросту "занудой". Но анализ некоторых не вполне обыденных ситуаций показывает, что над ними стоит все же поразмыслить.

Общение с компьютером и шлифовка языка

Итак, общение с компьютером, развитие логического (математического) мышления вырабатывают привычку к четкости, точности, однозначности. Миллионы людей начинают задумываться над реалиями, которые, к сожалению, отнюдь не всегда соответствуют декларациям, в том числе и официальным.

Эти миллионы людей общаются с десятками миллионов других, не столь привыкших к четкости, точности, однозначности, неуклонному соответствию слов и поступков. Оказывается, понимание реальности правил и законов может улучшить условия существования индивидуума - и все больше людей задумываются над "отдельными", "нетипичными" несоответствиями, а порой и пытаются эти несоответствия устранять.

Компьютеризация охватывает все стороны нашей жизни. Все больше членов нашего общества ощущают удобства как от общения с компьютером, так и от перестройки своего мышления.

...Пещерный человек хорошо знал, чего ему следовало опасаться: с мамонтом не дерись, тигра остерегайся, змею к телу не допускай.

Укрощение огня не только позволило обогревать жилище и варить пищу, но заставило опасаться пожаров и ожогов.

Создание паровой машины не только позволило построить железную дорогу, но и заставило остерегаться быстродвижущихся агрегатов.

Развитие химии и химической технологии не только обеспечило нас новыми лекарствами, новыми видами одежды, но и потребовало предохранения от попадания незнакомых веществ в пищу и на кожу.

Широкая электрификация вместе с возможностью использования электричества потребовала и защиты от поражения электрическим током.

Использование атомной энергии потребовало новой, особенной защиты.

Компьютер усиливает интеллектуальные возможности человеческого мозга. Но как опасно соприкосновение незащищенной руки с электрическим током, так и в обществе, где царят непродуманные, невсеохватывающие, нередко даже противоречащие друг другу (а потому невыполняющиеся) законы, применение компьютера принесет лишь новые трудности.

Автор считает, что законы, правила и другие нормы необходимо в предверии компьютерной эры пересмотреть: сформулировать их строго однозначно, непротиворечиво, не допуская двусмысленного толкования.

Необходимо повышать культуру языка - и не только законодателей, которые должны владеть языком безупречно, но и большинства населения. ■

Анонс

"Монитор" - версия 2.92

Ю.Крючков. Опыт работы с Dr.DOS версии 5.0.
Особенности работы операционной системы Dr.Dos 5.0 фирмы Digital Research. Методы настройки системы и максимального освобождения нижних разделов памяти. Обеспечение различных уровней доступа к информации с помощью паролей. Совместимость распространенных программных продуктов с Dr.DOS 5.0.

А.Чернышков. Clipper 5.0 - новые возможности (часть 2).
Классы памяти и типы данных в Clipper 5.0.

Д.Фивейский. Инструментальные средства для Clipper.
Описание отечественных средств для разработки прикладных программ в среде Clipper 5.0.

К.Белянский. Система текстовых окон на языке C++.
Реализация простого оконного интерфейса на языке C++. Основное внимание уделено средствам объектно-ориентированного программирования и новым синтаксическим конструкциям языка C++.

М.Остринский. Обзор графических редакторов.
Сравнение возможностей четырех популярных графических редакторов (Paint Brush 4+, PC Illustrator, Dr_Halo III, Picture Maker).

К.Шереметьев. Форт - язык для суперпрофессионалов.
Профессионалу нужен свой личный инструмент, отлично вычищенный и смазанный, всегда готовый к реализации любой идеи. Ему нужен язык, позволяющий разрабатывать свой язык. Это - язык для суперпрофессионалов, язык четвертого поколения, язык Форт. Форт, который позволяет оставить костыли ключевых слов и придуманных заранее аргументов и заставляет полагаться только на себя.

Е.Равич. Выходите на связь (часть 2).
Подробно описывается порядок работы с HAYES-совместимыми модемами и их система команд (HAYES Extended Command Set).

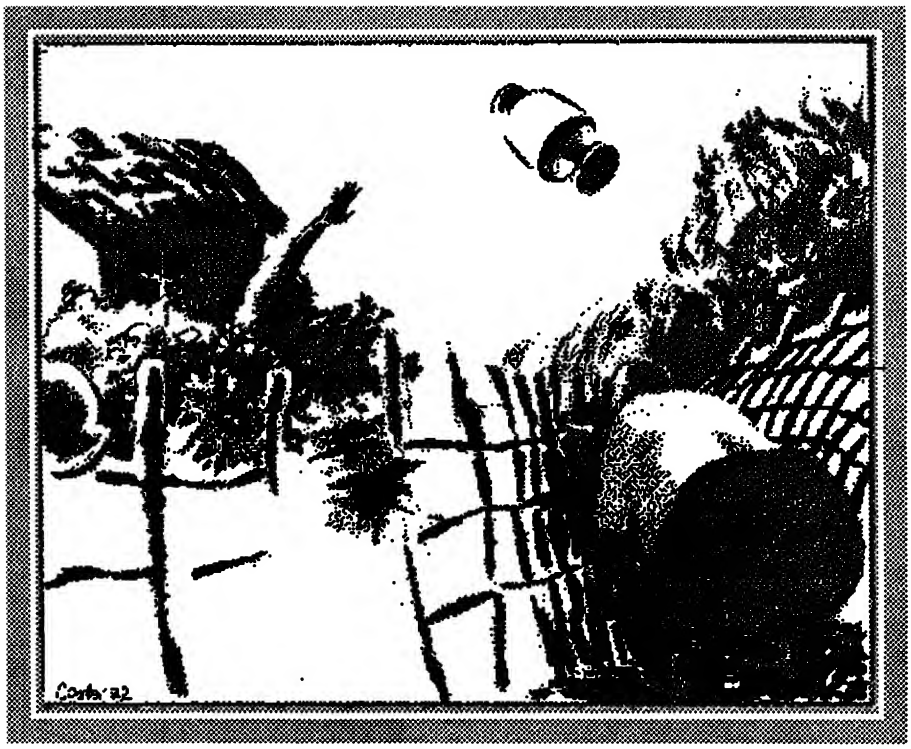
В.Аношин. Броня для файлов.
Рассказывается о том, как защищать исполняемые файлы от копирования. Приводится пример программы на языке C, реализующей простую защиту.

А.Спесивцев и др. Анализ отечественных систем защиты программ от копирования (часть 1).
В первой части статьи рассказывается о структуре и функциях систем защиты от копирования и разбираются недостатки некоторых существующих систем защиты. Приводятся рекомендации для пользователя - на что следует обращать внимание при выборе системы защиты.

Статьи по бизнесу, праву, зарубежная компьютерная хроника и другие материалы.

Про то, как фирма IBM с фирмой Microsoft поссорилась

А. Ларионов



Ты мне больше не дружок...

Где-то к середине 1991 года отношения между фирмами IBM и Microsoft,которые в течение 10 лет были партнерскими, стали походить на полномасштабную войну. В качестве фрагментов сводки боевых действий приведем следующие факты. Недавно Microsoft продемонстрировала программу,которая по утверждению этой фирмы, будучи запущенной в операционной системе OS/2 (сейчас дальнейшим развитием этой системы занимается только фирма IBM), может физически вывести из строя монитор персонального компьютера IBM PS/2. В свою очередь IBM представила программу, которая приводит к фатальному сбою системы Windows (программный продукт Microsoft), тем самым пытаясь продемонстрировать, насколько нестабильна эта система. Напомним,что первая версия операционной системы OS/2, выпущенная несколько лет назад, являлась совместной разработкой фирм Microsoft и IBM. Система оказалась неудачной. Сейчас Microsoft утверждает,что если OS/2 хоть как-то работает, то лишь потому, что в ней есть здоровая часть, созданная самой этой фирмой. Та же часть этой системы,которая сделана программистами IBM, фирмой Microsoft сейчас всячески поносится, так же, как и новые версии OS/2. В свою очередь программисты из IBM утверждают, что тот про-

Дело, конечно же, не в том, чтобы просто встать в этой борьбе на чью-либо сторону. Главное – выбрать программное обеспечение для такой операционной системы, на которую можно было бы положиться в будущем

граммный код,который они получили от Microsoft в процессе совместной работы над проектом по OS/2,годится лишь для того,чтобы отправить его на свалку.

Фирма IBM и конец света

В среде американских разработчиков программного обеспечения сейчас даже появился анекдот на тему взаимоотношений двух титанов компьютерной индустрии . Господь Бог вызвал к себе Президента США Джорджа Буша и президента Microsoft Билла Гейтса и объявил им,что он решил через 25 дней устроить конец света. Вернувшись в страну, Буш обратился по национальному телевидению к американцам и сказал : "У меня есть для вас две новости, одна хорошая, другая плохая. Вначале хорошая. Мы,американцы,были правы, когда обращались в своих помыслах к богу. Бог действительно существует. Плохая же новость состоит в том,что Бог решил устроить через 25 дней конец света." Билл Гейтс собрал правление своей компании и объявил : "У меня есть для вас две превосходные новости . Первая состоит в том,что существует Господь Бог . Вторая новость заключается в том,что через 25 дней он уничтожит нашего главного противника – IBM."

Нерушимый блок IBM и Microsoft

Ситуация в индустрии программного обеспечения для персональных компьютеров всегда отличалась изменчивостью, но скорость, с которой здесь произошли изменения за 1991 год, пожалуй, не имеет прецедентов. В ноябре прошлого года IBM и Microsoft выпустили совместное заявление,касающееся разделения рынка настольных персональных компьютеров на три уровня : нижний уровень – MS DOS, средний уровень – Windows, верхний уровень – OS/2. В январе 1991 года Microsoft провела семинар для аналитиков и для прессы по стратегическим направлениям деятельности фирмы в области операционных систем с целью провозглашения своей долгосрочной политики в этой области. На этом семинаре вице-президент Microsoft Стив Боллер утверждал, что OS/2 отнюдь не является мертвым или покинутым проектом для его фирмы. В числе приоритетных программ фирмы были упомянуты DOS,Windows и OS/2. Отношения же с IBM Стив Боллер охарактеризовал,как "здоровое и долгосрочное сотрудничество".

Завтра была война

Вскоре после этого между IBM и Microsoft разразилась война. И в этом одна из причин того,что ближе к середине 1991 года стратегия Microsoft претерпела заметные изменения. Технические аспекты этого таковы. Наибольшие изменения коснулись версии OS/2 3.0,над которой, как предполагалось, обе фирмы будут работать совместно. В недавних брифингах для прессы официальные представители Microsoft дали понять,что интерес их фирмы к разработке следующих версий OS/2 почти что равен нулю. С другой стороны,представители IBM заявляют ,что им дела нет до того,что Microsoft думает по этому поводу. Началом войны можно считать конец зимы 1991 года, когда прогремел первый залп со стороны Microsoft, которая объявила, что не будет участвовать в дальнейшей работе над OS/2 и сконцентрирует свои усилия в работе над Windows, которую она считает своим наиболее жизнеспособным продуктом в обозримом будущем. Затем в апреле последовал ответ IBM, которая заявила,что и без участия в работе Microsoft она добьется того,что OS/2 будет настолько хорошо имитировать работу MS DOS,что будет лучшей DOS, чем сама DOS. Аналогичное утверждение было сделано в отношении системы Windows. Многие аналитики рассматривают неожиданный союз между фирмами IBM и Apple,как еще одну контратаку на Microsoft. Последние признаки этикета во взаимоотношениях IBM и Microsoft бесследно испарились,когда в июле Стив Боллер продемонстрировал программу, созданную Microsoft специально для "подвешивания" OS/2. Тогда же Боллер поставил под сомнение способность IBM самостоятельно поставить новую версию OS/2 и более того, выразил некоторое недоумение по поводу того, зачем вообще нужна операционная система OS/2. По словам Боллера, не слишком высокие потребности вполне покрываются MS DOS и Windows, а более изощренные запросы будут удовлетворены создаваемой сейчас системой Windows NT. "OS/2 – это не рыба и не мясо. Эта система застряла где-то посередине. Не существует ничего,что бы она делала достаточно хорошо" – заявил Боллер.

Но идея-то была красивой !

В 1988 году IBM и Microsoft представили свой совместный программный продукт – операционную систему OS/2, разрекламированную как преемник MS DOS. OS/2,как было обещано,явится воплощением мечты любого пользователя, привыкшего к работе с MS DOS : она будет способна обеспечивать наиболее продвинутые возможности,такие как расширяемость,мультизадачность и способность работать с компьютерными сетями. Но обещанная в рекламных проспектах мечта обернулась в реальной жизни настоящим кошмаром. Первая версия OS/2 была очень плохо совместима с MS DOS, имела неудобные средства инсталляции и требовала специальной адаптации для очень многих компьютерных систем. За три года с момента появления на рынке программного обеспечения этой операционной системы было продано около четырехсот тысяч ее экземпляров. Для сравнения укажем,что за 10 лет существования операционной системы MS DOS было продано около семидесяти миллионов экземпляров легально и огромное количество было скопировано незаконно, так что на основании этих цифр можно с уверенностью сказать о том,что фактичеки OS/2 провалилась. Вскоре после этого Microsoft стала проталкивать на рынок свою собственную разработку – систему Windows, с момента появления которой на рынке было продано около четырех миллионов ее экземпляров. Именно из-за указанных выше обстоятельств на гладкой поверхности отношений этих фирм появилась первая трещина, кото-

рая в этом году превратилась в глубокую пропасть.

Немного терпения, и ваша щетина превратится в золото !

Ответом IBM на провал первой версии OS/2 стало заявление следующего содержания : "Забудьте все проблемы и неудобства с OS/2 первой версии. По-настоящему мощная вторая версия этой системы компенсирует вам все издержки работы с первой". Однако неучастие Microsoft в работе над следующими версиями OS/2 делает вышеупомянутое обещание для IBM весьма трудно выполнимым, и пока фирме не удалось существенно продвинуться в этой области. Достигнутый уровень поддержки Windows системой OS/2 совершенно недостаточен. Некоторые специалисты указывают на то,что IBM в течение 10 лет находилась в роли пассажира в автомобиле, который вела Microsoft, и теперь, когда ведущего в машине нет, IBM осталась один на один со своими проблемами, и в связи с этим выражают сомнение в способности фирмы самостоятельно создать конкурентноспособное программное обеспечение. Пока что IBM выпустила новую версию OS/2, снабженную более удобными средствами для инсталляции, добавив туда графическую инсталляционую программу и возможность выбора операционной системы (DOS , OS/2 1.3 , OS/2 2.0) после включения компьютера, а также программу, обучающую работе со средствами тестирования инсталяционных программ. Тем не менее,эта версия не поддерживает в полном объеме Windows, а ведь отправной точкой в работе над следующими версиями OS/2 для IBM было обещание сделать OS/2 еще лучшей Windows,чем сама система Windows. Тем не менее до конца этого года фирма обещает сдержать свое слово.

Наших планов громадье ...

Microsoft объявила, что то, что раньше именовалось OS/2 3.0, будет называться Windows NT, и эта система вообще не будет иметь ничего общего с OS/2 . Так что Windows NT теперь будет конкурентом всего того, что планируется фирмой IBM в отношении новых версий системы OS/2, а отнюдь не их составными частями, как это предполагалось в конце прошлого года. Все программное обеспечение для нынешней версии системы Windows и существующие прикладные программы для Windows и DOS для 16-разрядных ЭВМ будут работать под управлением системы Windows NT. По словам Боллера, операционная система Windows NT будет требовать для своей работы около восьми мегабайт оперативной памяти и будет нацелена на использование быстрых 386 и 486 микропроцессоров, а также микропроцессоров нового поколения типа RISC. Для пользователей настольных персональных компьютеров со средними возможностями Microsoft выпустит версию Windows для DOS 6.0, рассчитанную на 32-разрядные компьютеры,но не раньше 1993 года.

programmer

Почти все программное обеспечение DOS полностью поддерживается Windows

Кому вершки,а кому корешки

Разногласия между IBM и Microsoft вызваны не только разницей во взглядах на основные направления компьютерной технологии, но еще и борьбой за влияние в сфере бизнеса персональных компьютеров. Microsoft, как и подобает довольно амбициозной фирме, хотела бы контролировать направления дальнейшего развития операционных систем для персональных компьютеров, но к этому стремится также и фирма IBM. Хотя IBM скорее всего выпустит следующую версию OS/2, но без поддержки Microsoft как в разработке системы, так и ее внедрении,новую версию, по-видимому, тоже ожидает неудача. Если прошлое чему-то научило пользователей персональных компьютеров, так это скорее всего тому, что не следует вкладывать ни гроша в OS/2. Сейчас IBM испытывает такие трудности с приобретением прикладных программ для OS/2,что она даже объявила о намерениии предложить деньги тем добровольцам, кто создаст какие-нибудь прикладные программы для второй версии OS/2. Пока что желающих немного и, кроме того, остается вопрос о том, можно ли будет перенести эти программы в новую операционную систему, создание которой на принципах объектно-ориентированного программирования предусматривается совместным проектом под названием Pink фирмами IBM и Apple. Пока что ни одна из фирм - производителей персональных компьютеров не заявила о поддержке этого проекта. Напротив, система Windows сразу привлекла к себе внимание разработчиков программного обеспечения без всяких денежных посулов. В конце прошлого года Microsoft выпустила пакет инструментальных средств для разработки программного обеспечения для Windows NT (Windows NT Software Development Kit, сокращенно SDK). SDK поддерживет как компьютерные системы, использующие процессор фирмы Intel, так и системы, использующие улучшенный стандарт компьютерных сред (Advanced Computing Environment,сокращенно ACE). Поскольку более 60 компаний-производителей компьютеров заявили о поддержке ACE, разработчики прикладных программ для Windows NT могут чувствовать уверенность в том,что рынок компьютеров, использующих этот стандарт, будет достаточно большим.

Война – ничто, а главное – маневры

Пока что борьба между IBM и Microsoft за влияние на будущие поколения операционных систем не имеет очень большого значения для отдельных пользователей. В самом деле, обе враждующие стороны обещают, что прикладные программы для DOS и Windows будут работать независимо от того, какую операционную систему они создадут

в будущем. Для разработчиков программного обеспечения ситуация сложнее. Недружественные акции двух столпов мира персональных компьютеров по отношению друг к другу отнюдь не создают стабильной обстановки, необходимой для долгосрочного планирования. По поводу нынешней враждебности IBM и Microsoft существует одно довольно интересное суждение, которое трактует все происходящее, как развивающееся согласно сценарию "Война – ничто, а главное – маневры". Согласно этой точке зрения, обе фирмы придумали весьма удобное средство – игру в войну для того, чтобы держать внимание компьютерной индустрии на себе и своих основных программных продуктах. Названия этих фирм постоянно находятся в заголовках новостей, что повышает интерес как пользователей, так и разработчиков программ и аппаратуры к продукции обеих фирм. Хотя, честно говоря, мне трудно представить себе, чтобы президенты двух этих фирм измыслили такую хитроумную стратегию для проталкивания своей продукции на рынке, и такая подоплека событий представляется маловероятной. Вообще говоря, никто пока не мешает IBM и Microsoft помириться. Дело в том, что вряд ли когда-нибудь операционной системе OS/2 удастся захватить значительную долю персональных компьютеров (скорее всего эта доля не будет превышать 15 - 20%) из-за слишком высоких требований, предъявляемых к ресурсам компьютера, в частности к объему оперативной памяти.

На западном фронте без перемен

Но пока что нет никаких признаков примирения. Microsoft проталкивает Windows. IBM старается навязать пользователям OS/2. Конечно, предлагаемые продукты на рынке персональных компьютеров этим не исчерпываются. Апологеты операционной системы Unix призывают к тому, чтобы отказаться и от Windows и от OS/2 и предпочесть Unix. Фирма Apple предлагает выбрать Macintosh в качестве вашего будущего персонального компьютера. Продавцы программного обеспечения, вовлеченные в инициативу распространения ACE, стараются убедить, что лучше расстаться с системами, основанными на архитектуре микропроцессора Intel, и взять за основу архитектуру MIPS. Мнения относительно того, какая альтернатива возьмет верх, очень сильно разнятся. В данной ситуации лучшим советом, по-видимому,будет следующий. Не принимать далеко идущих решений и не отдавать обязывающих предпочтений какой-либо фирме, пока это не станет абсолютно необходимым. Пока в обещаниях создателей операционных систем так много неопределенного, разумной политикой является возможность иметь как можно более широкое поле выбора. Это, конечно, не означает впадания в состояние абсолютной бездеятельности. Если надо сделать выбор для выполнения определенной работы, то его надо сделать. Речь просто идет о том, чтобы не делать выбор, очень жестко связанный с конкретными особенностями стратегии данного производителя операционной системы. В некоторых случаях, возможно, имеет смысл вообще отложить принятие решения до тех пор, пока не станет ясно, каков наиболее вероятный исход войны. Следует сопоставить риск последовать стратегии фирмы, от которой она может позже отказаться, со стоимостью отсрочки применения новых систем. Еще один уместный в данной ситуации совет содержится в старой пословице о том, что лучше синица в руках, чем журавль в облаках. Безотносительно к всевозможным обещаниям разных производителей программных продуктов, безотносительно к тому, что написано в разных статьях (в том числе и в этой), программное обеспечение, которое вы не можете сами запустить и проверить, как оно работатет, не идет по значимости для вас ни в какое сравнение с тем, с которым вы можете все это проделать.

> user
>
> Сейчас борьба между IBM и Microsoft за влияние на будущие поколения ОС не имеет очень большого значения для отдельных пользователей

Вот наступит время,время всех рассудит

К счастью, мы уже достаточно близки к тому моменту времени, когда ситуация с тремя ключевыми программными продуктами окончательно прояснится. Как было обещано, OS/2 2.0 (IBM),Windows 3.1 и SDK (Microsoft) будут готовы к концу 1991 года. Так что уже в начале 1992 года мы будем иметь вместо многочисленных прогнозов реальные факты. Тогда мы узнаем, кто выполняет обещания относительно своих программных продуктов, а кто нет. Что еще более существенно, мы будем иметь возможность непосредственно проверить, что изготовили IBM и Microsoft и выяснить для себя, какая из фирм способна обеспечить прочную основу для разработки следующих систем. Дело, конечно же, не в том, чтобы просто встать в этой борьбе на чью-либо сторону. Главное – выбрать программное обеспечение для такой операционной системы, на которую можно было бы положиться в будущем. Вскоре это можно будет сделать не на основе обещаний, а на основе опыта работы с продуктами, которые мы сможем установить и проверить на прикладных программах. Все это, конечно, при условии, что обещаниям создателей операционных систем вообще можно верить. Но, вероятно, все-таки следует подождать. ■

Обзор подготовлен по материалам журнала "PC Week".

Рынок программных продуктов в России: первые шаги

Д.Бройтман

Clipper расправляет паруса

С 3 по 6 ноября 1991 года в Обнинске проходила первая конференция фирмы Nantucket в России. В ее работе участвовали более 300 зарегистрированных пользователей Clipper 5.0 - системы разработки прикладных программ, предназначенной для профессиональных программистов. Организовало конференцию СП Магнит – официальный дистрибьютор фирмы Nantucket в нашей стране.

Nantucket является на сегодняшний день, по-видимому, единственной западной фирмой, которая предоставляет своим пользователям в России такой же уровень обслуживания, как и своим пользователям в других странах (продажа за рубли, документация на русском языке, поддержка по "горячей" телефонной линии, личные консультации в центре технической поддержки в Москве, распространение журнала "Nantucket News" на русском языке, скидки при приобретении новых версий и т.д.).

Адрес СП "Магнит" : 127018, Москва, 2-я Ямская ул., д.15.

Телефоны : (095) 289-44-77, (095) 289-44-83

Software Development–92

С 27 февраля по 1 марта 1992 года в Москве на ВДНХ будет проходить выставка программного обеспечения Software Development - 92. Основная цель проводимых в рамках выставки мероприятий – представление работ лучших программистских коллективов страны предпринимателям и пользователям. Организаторы выставки планируют привлечь к участию в ее работе представителей западных фирм

▶*начало на стр. 40*

Зарубежная компьютерная хроника

1991 год, декабрь

Немецкий электрический и электронный концерн Siemens объявил 12 декабря, что он владеет 95% акций убыточной компьютерной компании Nixdorf и может теперь полностью поглотить ее.

Карло де Бенедетти, руководитель итальянской фирмы Olivetti, подтвердил, выступая 12 декабря в парламентской комиссии по промышленности, что Olivetti ведет переговоры с французской компанией Cap Gemini Sagetti, ведущим поставщиком программного обеспечения в Европе.

Olivetti традиционно сильна в производстве аппаратуры, однако насыщенность мирового рынка привела к тому, что потери фирмы в первой половине года составили 73,7 млрд. итальянских лир (60,5 млн. долларов).

Ухудшающееся положение фирмы вынудило де Бенедетти вновь принять руководство ее деятельностью, и 29 ноября он произвел радикальную перестройку подразделений Olivetti.

Численность персонала фирмы уменьшилась в этом году с 53679 до 47000.

Акции корпорации Digital Equipment, второго по величине производителя компьютеров в мире, упали 18 декабря на 6 долларов: до 50 1/2 долл. за акцию.

Спрос на большие компьютеры фирмы DEC резко понизился из-за экономического спада, охватившего всю мировую промышленность, и усиления конкуренции.

Производство больших машин традиционно было одним из самых прибыльных сегментов компьютерного рынка.

В конце финансового года (в июне месяце) Digital сообщила о первых в своей истории годовых потерях в 617 млн. долларов и о проведении серьезной внутренней перестройки.

Digital также сообщила о том, что она планирует уменьшить численность работающих на 10000 человек.

Фирма Advanced Micro Devices 19 декабря объявила, что к концу года она продаст 2 млн. производимых ее кристаллов 386-го микропроцессора. Акции AMD выросли в этот день на 1 5/8 долл.: до 16 5/8 долл. В то же время стоимость акции фирмы Intel упала с 44 7/8 долл. до 42 3/4 долл.

Неожиданно высокий спрос на микропроцессор Am386DX и сохраняющийся высокий спрос на Am386SX привели к росту дохода от продаж устройств семейства Am386 до 120 млн. долл. в текущем квартале, что более чем вдвое превысило доход прошлого квартала (59 млн. долл.).

В то же время Intel расширяет продажу 486-х микропроцессоров следующего поколения и готовится к выпуску более мощной 586-й версии, рассчитывая сохранить свое лидерство.

Intel оспаривает право AMD на выпуск спроектированных Intel микропроцессоров. Назначенный компаниями для разрешения этого спора третейский судья должен вынести окончательное решение в ближайшее время.

с тем, чтобы способствовать развитию экспорта отечественных программных продуктов и компьютерных технологий.

В программу выставки помимо постоянно действующей экспозиции входят: профессиональные семинары по программированию в современных средах; семинары фирм, участвующих в выставке; презентация наиболее интересных работ индивидуальных авторов.

Контактный телефон : (095) 261-97-58
Факс : (095) 257-05-29

Поставщики программ объединяются

5 декабря 1991 года в Москве состоялось первое собрание Ассоциации поставщиков программных продуктов (АППП).

В число учредителей Ассоциации входят крупнейшие отечественные и зарубежные фирмы - поставщики программных продуктов: Lotus, Microsoft, Nantucket, СП ПараГраф, СП Магнит, СП ИнтерМикро и другие.

АППП регистрируется как общественная организация и не ставит своей целью извлечение прибыли. АППП является ассоциацией фирм. Основной орган - конференция, на которой каждая фирма имеет один голос. Предполагается, что президентом АППП станет Роберт Клафф (фирма Nantucket).

Цели ассоциации:

• сбор и распространение информации о рынке программных продуктов;

• борьба с нелегальным распространением программного обеспечения и защита авторских прав программистов;

• обучение пользователей и дилеров;

• взаимодействие с государственными органами с целью принятия разумного законодательства по авторским правам и интеллектуальной собственности.

Контактный телефон : (095) 289-44-83

Мастер - не ломастер

С 9 по 13 декабря 1991 года в Москве проходила школа-семинар по использованию Мастер-технологии. В ее работе участвовали более 100 пользователей Мастера из разных городов.

В настоящее время все силы, работающие над Мастер-технологией, во главе с автором Мастера и Лексикона Е.Н.Веселовым, сосредоточились в СП Микроинформ. Разработчики Мастера надеются, что их продукт займет среди отечественных программных средств место, аналогичное тому, которое занимают на западном рынке такие системы, как Lotus 1-2-3 и dBase.

Во вступительном докладе Е.Н.Веселов сообщил, что через несколько месяцев будут доступны новые версии системы (условные названия - Мастер-2 и Лексикон-2), выполненные на современном технологическом уровне. Мастер превратится из замкнутого программного модуля в открытую технологию (набор библиотек, утилит, концепций, язык программирования), позволяющую строить прикладные системы. Технология будет открыта для добавления модулей на Си. Мастер будет реализован в DOS, Windows и других оконных системах и будет обеспечивать со-

Зарубежная компьютерная хроника

1992 год, январь

Согласно данным Software Publishers Association и Business Software Alliance потери американских поставщиков от компьютерного пиратства в Западной Европе составили в 1990 году, по самым скромным подсчетам, 4,46 млрд. долларов.

Эти две организации представляют крупнейшие компании - производители программного обеспечения для ПК, большинство из которых расположено в США. Оценки потерь основаны на сопоставлении количества проданных копий программ с количеством проданных персональных компьютеров.

Потери от пиратства в самих США в том же году оцениваются в 2,4 млрд. долларов.

Японская электронная компания Fujitsu и британская фирма - производитель компьютеров ICL ведут переговоры о возможной продаже производимых ICL IBM-совместимых персональных компьютеров в Японии.

На рынке персональных компьютеров в Японии доминирует фирма NEC. Она и другие японские фирмы - производители компьютеров продают в Японии ПК, несовместимые с преобладающими на мировом рынке ПК стандарта MS DOS.

Американская фирма AT&T и японская NEC объявили о начале совместной работы по проектированию и производству микросхем статической памяти с произвольным доступом (S-RAM) емкостью до 4Мбит.

Микросхемы S-RAM потребляют меньше энергии по сравнению с более распространенным типом памяти - D-RAM. Микросхемы будут производиться NEC и продаваться обеими компаниями.

Соглашение между NEC и AT&T так же, как и несколько других, заключенных в последнее время между японскими и американскими полупроводниковыми компаниями, ставит целью смягчение торговых трений между двумя странами.

---

10 крупнейших в мире поставщиков полупроводникового оборудования (по данным VLSI Research)

1980 год		1990 год	
Perkin-Elmer	США	Tokyo Electron	Япония
GCA	США	Nikon	Япония
Applied Materials	США	Applied Materials	США
Fairchild	США	Advantest	Япония
Varian	США	Canon	Япония
Teradyne	США	Hitachi	Япония
Eaton	США	General Signal	США
General Signal	США	Varian	США
Kulicke & Softa	США	Teradyne	США
Takeda Ryken	Япония	SVG	США

вместимость в направлении DOS - Windows.

Основная концепция разработки прикладных систем в Мастере - концепция визуальной раскрутки (система строится внутри самой себя).

Вводится понятие классов сред с наследованием свойств, объектной ориентированностью и событийной управляемостью.

В качестве встроенной модели данных выбрана модель данных SQL.

Серьезные изменения претерпит текстовый редактор Лексикон. Модель текста будет усовершенствована в соответствии с возросшими требованиями к обработке текстовой информации. Лексикон будет строится как приложение в новой Мастер-технологии. Он станет не просто текстовым редактором, но и средством разработки текстового интерфейса для прикладных систем.

Наступление на пиратов

15 - 17 декабря 1991 года в Москве прошла конференция по защите авторских прав программистов. На конференцию из разных городов съехались представители фирм - поставщиков программных продуктов, пользователи, программисты, юристы.

Нелегальное копирование программ для личных целей и пиратство, т.е. незаконная перепродажа программ с целью получения прибыли, приводят к подрыву отечественной индустрии программного обеспечения. Для успешной борьбы с пиратством, по мнению участников конференции, необходим комплекс юридических мер: принятие в России законодательства по охране авторских прав, соответствующего Европейской Директиве по охране компьютерных программ (декабрь 1990 года), присоединение России к Бернской конвенции по охране авторских прав и к Генеральному соглашению по тарифам и торговле (под него подпадает экспорт и импорт любых товаров, в которых заключена интеллектуальная собственность).

Кроме этого, ведущие фирмы - поставщики программ, действующие в нашей стране, намерены проводить согласованную политику в борьбе с пиратством: демонстрировать пользователям выгоды легального приобретения программ, предавать огласке случаи наиболее злостного пиратства, создавать неблагоприятное для пиратов общественное мнение.

Примером для нас могут служить страны Юго-Восточной Азии (Сингапур, Таиланд, Малайзия), где еще несколько лет назад уровень пиратства был чрезвычайно высок. Работа ассоциаций фирм - поставщиков ПО привела к появлению в этих странах развитого законодательства, существенному сокращению пиратства и, как следствие, к бурному росту собственной индустрии программного обеспечения. ■

▶ *начало на стр. 40,58-59*

Зарубежная компьютерная хроника

Судебные дела по обвинению в воровстве программного обеспечения, возбужденные в Великобритании ведущими американскими поставщиками ПО против Mirror Group Newspapers и Совета лондонского района Гринвич, отзываются из суда. Совет Гринвича объявил 10 января, что он согласен уничтожить нелегальные копии программ, купить их у дилеров, выплатить поставщикам ПО компенсацию за нелегальное использование программ и оплатить судебные издержки. Подобное заявление ожидается и от Mirror Group Newspapers.

---

13 января фирма Hewlett-Packard объявила о выпуске новой рабочей станции стоимостью 5000 долл. Ее производительность примерно на 75% выше, чем производительность аналогичных по стоимости рабочих станций фирм Sun и Digital.

В настоящее время Sun удерживает около 40% рынка рабочих станций, а Hewlett-Packard занимает второе место - 16%.

Японское министерство международной торговли и промышленности обратилось к 226 японским компаниям, составляющим около 90% японского рынка полупроводников, с просьбой покупать по меньшей мере 20% полупроводников у иностранных поставщиков.

В июле между Японией и США было заключено соглашение, по которому доля иностранных производителей на японском рынке полупроводников должна составлять не менее 20%. Реально в четвертом квартале 1991 года она составляла 14 - 16 %.

Sun Microsystems, ведущий производитель рабочих станций, и Cray Research, лидер в производстве суперкомпьютеров, договорились о сотрудничестве. Cray будет использовать при проектировании компьютеров сверхвысокой производительности технологию, основанную на сокращенном наборе команд SPARC, который фирма Sun применяет в своих RISC микропроцессорах.

Европейский рынок полупроводников (предварительные оценки Dataquest)

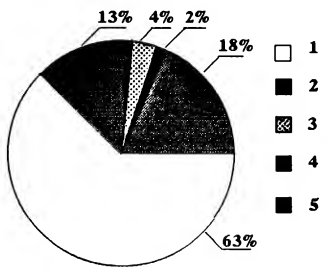
Фирма	Место в 1990г.	Место в 1991г.	Продажи в 1990г. ($ млн)	Продажи в 1991г. ($ млн)	Доля рынка в 1991г. (%)
Philips	1	1	1,155	1,172	10,3
Siemens	2	2	964	958	8,4
SGS-Thomson	3	3	908	887	7,8
Motorola	4	4	765	770	6,8
Intel	6	5	622	760	6,7
Texas Instruments	5	6	637	629	5,5
Toshiba	7	7	490	509	4,5
NEC	8	8	417	452	4,0
National Semiconductor	9	9	408	408	3,6
Hitachi	10	10	273	318	2,8

По материалам журнала "Economist" и газеты "Financial Times"

Итоги торгов Компьютерной секции Московской товарной биржи в ноябре – декабре 1991 года

С.Воробьев, С.Лаптев

Компьютерная секция – это самоуправляемое структурное подразделение МТБ, объединяющее коммерческие фирмы, специализирующиеся на оптовой торговле вычислительной техникой, оргтехникой, средствами связи, бытовой электроникой. На протяжении ноября - декабря 1991 года торги на Компьютерной секции проводились регулярно два раза в неделю и объем сделок был достаточно стабильным с ярко выраженной тенденцией к росту. Общий объем продаж товаров секции в декабре достиг 80 млн. рублей, что в 2 раза выше ноябрьского уровня. Доля оборота секции в общем объеме продаж Московской товарной биржи по сравнению с ноябрьским уровнем в декабре возросла на 3.5 % и составила 7.9 %. Структура оборота секции очень динамична. В первую очередь она зависит от эффективности торговли по группам товаров. В ноябре 53 % оборота секции приходилось на видеотехнику, в декабре эта доля возросла до 63%. Необходимо отметить и увеличение в общей структуре



1. Видеотехника
2. Факсы, телефоны
3. Множительная техника
4. Электроника без спецификации
5. Компьютеры

Структура оборота Компьютерной секции МТБ в декабре 1991 года

МОНИТОР-1992

Подписной купон

МОНИТОР

1	2	3	4	5	6	7	8	9

/

Адрес:_____

_____

Ф.И.О._____

Стоимость подписки на один номер журнала – 15 рублей (действительна до 01.05.92)

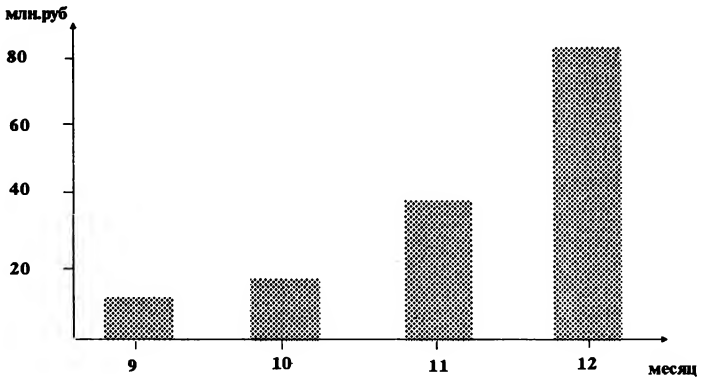
Отметьте крестиками те номера журнала, которые Вы желаете получить, укажите свой адрес и фамилию.

Перечислите сумму оплаты на счет НТО "Софт-Москва": р/с 467002 в к/б "Банк развития XXI век ", к/с 161816 в РКЦ ГУ ЦБ РСФСР по г.Москве, МФО 201791.

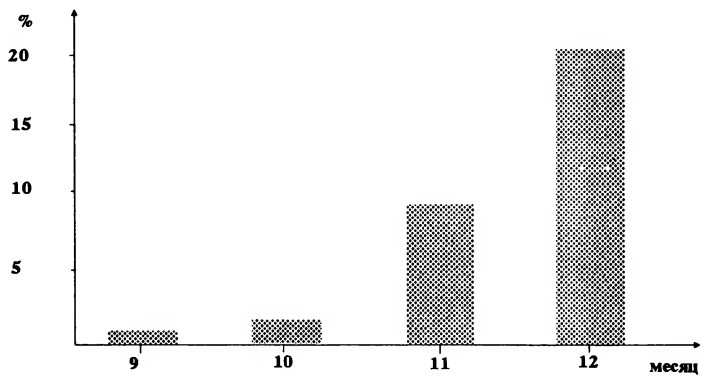
Вышлите купон и квитанцию об оплате по адресу: 117419, Москва, а/я 765.

оборота доли компьютеров и периферийного оборудования на 10 %. В среднем цены на вычислительную технику возросли в декабре на 80 %. Оптовая цена комплекта IBM PC /AT 286/287 стандартной конфигурации в начале декабря составляла 110-120 тыс. рублей, а к концу месяца возросла до 180-190 тыс. рублей. Объективных причин роста цен на вычислительную технику немало: это и инфляционное ожидание продавцов, и угрожающий уровень разбалансированности рынка вычислительной техники, а также определенная налоговая и таможенная политика российского правительства. Начавшийся 2 января процесс либерализации цен (оказавшийся пока в большинстве случаев просто многократным повышением цен на все товары) отразился не только на розничной торговле, где цены в среднем поднялись в 10 раз, но и на биржевой оптовой торговле, где скачок цен был безусловно меньше по сравнению с розничными. В течение последних месяцев все рыночные структуры жили в ожидании. Введение свободных цен, перспектива развития российского предпринимательства в условиях распада единого рублевого пространства, неопределенность финансового состояния большего числа коммерческих и государственных структур в рамках нового налогового российского законодальства стали определяющими факторами снижения деловой активности во всех секторах экономики.

Контактный телефон: (095) 188-76-83 ■



Динамика общего оборота Компьютерной секции (сентябрь–декабрь 1991 года)



Динамика продаж компьютеров в общем объеме продаж товаров Компьютерной секции

МОНИТОР-ДИСК-1992

Подписной купон

МОНИТОР

1	2	3

Адрес:_____

_____

Ф.И.О._____

Журнал "Монитор" объявляет подписку на электронное приложение "Монитор-Диск". Каждая дискета будет содержать полные тексты программ из 3 (или 4) номеров журнала, а также некоторые неопубликованные в журнале материалы.

Стоимость подписки на одну дискету – 80 рублей (действительна до 01.05.92).

Отметьте крестиками те номера дискет, которые Вы желаете получить, укажите свой адрес и фамилию.

Перечислите сумму оплаты на счет НТО "Софт-Москва": р/с 467002 в к/б "Банк развития XXI век ", к/с 161816 в РКЦ ГУ ЦБ РСФСР по г.Москве, МФО 201791.

Вышлите купон и квитанцию об оплате по адресу: 117419, Москва, а/я 765.

Перестройка в халифате и в хозяйственном аппарате

А. Халамайзер

В одном арабском халифате на седьмом году перестройки опять свели бюджет с дефицитом. И заем, взятый у султана турецкого, не помог.

- Казна моя день ото дня пустеет, - пожаловался халиф своему визирю. - А ведь тысяча сборщиков по целому месяцу с верблюдов не слезают, из кишлака в кишлак мотаются, налоги собирают. Может, они только свою мошну набивают?

Выписал визирь из Турции бухгалтера-ревизора Мехмета-эфенди и назначил ревизию. Каждый двадцатый сборщик на утаенные деньги себе дворец построил, показала ревизия; еще трое из каждых двадцати по четыре неработающие жены содержат да челядь немалую.

Доложил визирь халифу. Тот повелел: пятьдесят великих казнокрадов на кол посадить, еще сто пятьдесят - без выходного пособия с волчьим паспортом уволить, по шесть соток земли выделить - пусть сами пшеницу да виноград сажают, сами овец разводят, а жены пусть сами лепешки пекут, сыр овечий готовят да вино давят - и тем семьи свои кормят-поят. За остальными - по два доверенных наблюдателя поставить, да сто над ними начальников - отчеты проверять. Наблюдателям двойную ставку назначить, начальникам - тройную. Чтобы на взятки не зарились.

Стало в казну куда больше денег поступать. Да вот беда: раньше по налоговому эмирату тысячу ставок выплачивали, а теперь с наблюдателями да начальниками - аж четыре тысячи с лишним. Дефицит бюджета опять вырос. Пришлось у султана новый заем просить, больше прежнего. Как дальше жить?

Выписал визирь из Турции консультанта-экономиста Ахмета-эфенди. Три месяца консультант к порядкам в халифате присматривался, а на четвертый пред светлые очи халифа предстал:

- Прикажи, о повелитель, собрать в тронном зале всех сборщиков, всех наблюдателей и начальников.

А когда собрались, расставил их от дверей до трона цепочкой. Халиф на трон уселся. Визирь по правую руку встал. Ахмет-эфенди по левую.

- Сейчас, о повелитель, принесут нам собранное со всего халифата масло, - объявил гость. Хлопнул в ладоши, двери отворились, и внесли служители огромный масляный шар. Твердый, прямо из государственного спецхолодильника. Велел Ахмет-эфенди передавать этот шар из рук в руки по цепочке.

Сборщики налогов передавали масло быстро; наблюдатели его в руках переворачивали, осматривали; а их начальники еще дольше вертели и только потом передавали.

Масло постепенно таяло. До рук визиря дошло меньше половины.

Хлопнул гость в ладоши - все взоры к нему обратились.

- Итак, о повелитель, все видели, что никто из присутствующих не взял себе ни кусочка масла. Но все видели также, что масла стало раза в три меньше. И это - в природе вещей.

Халиф ничего не смог возразить. Все остальные тоже молчали. Гость снова хлопнул в ладоши. И опять все взоры обратились к нему.

- Но это не только в природе вещей, о повелитель, но и в природе людей. Одни передавали масло быстро и не особенно измазали пальцы. Другие же поворачивали масло то так, то этак - и к их пальцам прилипло куда больше.

И увидел халиф, что многие присутствующие, ухмыляясь, облизывают пальцы.

Было это в одном арабском халифате полтысячи лет назад...

В Москве на научном симпозиуме директор специального конструкторского бюро встретился на седьмом году перестройки с директором хотя и не специального, но родственного бюро из Японии.

- Скажите, сколько инженеров в вашем бюро? - спросил москвич.

- Шестеро, - ответил японец. И рассказал, кто конкретно чем занимается. - А у вас?

Москвич смутился. В его бюро работали семьдесят инженеров, среди которых были "старшие", "ведущие", "персональные"... Но, чтобы не выглядеть идиотом, он решил слегка сократить это количество.

- А у нас семеро, - ответил он, не вдаваясь в детали.

На другое утро японец казался усталым. Москвич поинтересовался, хорошо ли ему спалось в гостинице "Россия".

- Всю ночь не сомкнул глаз, - вздохнул японец. - Все ломал голову: зачем вам седьмой инженер и чем бы он мог заниматься?

В этом конструкторском бюро, как и во всем хозяйственном аппарате, ни Мехмет, ни Ахмет пока еще не побывали. ■

MERISEL в Москве.
Время крупных посредников пришло

А. Свириденко

Рост рынка компьютерных технологий в России в последние годы вызвал существенное увеличение количества предложений на поставку аппаратуры и программ - главным образом от крупных производителей. По оценкам экспертов, примерно в середине лета 1991 года это привело к качественным изменениям в структуре рынка. Существенно усилилось дилерское звено, представленное внедренческими фирмами, магазинами и другими торговыми посредниками, появились оптовые фирмы, ориентирующиеся на закупку продукции непосредственно у производителей для ее реализации через собственных дилеров.

Эти структурные изменения на отечественном компьютерном рынке соотвстствуют современной мировой тенденции возрастания стратегической роли крупных оптовых посредников. В США, например, крупное оптовое посредничество зародилось в ранних 80-х, а в 1991 году, согласно данным журнала "Computer Reseller News", 61% всех покупок, связанных с компьютерными технологиями, проходил по каналам дистрибьютеров.

Отечественный же рынок компьютерных технологий крупных дистрибьютеров до сих пор не знал. Не знали мы и названия Merisel, Inc. – американской компании, являющейся одним из крупнейших в мире дистрибьютеров аппаратуры и мировым лидером в распространении программ с годовым оборотом более 1,5 млрд.долл.

Международную деятельность Merisel, штабквартира которой расположена в Лос-Анджелесе (Калифорния), проводит через свои дочерние фирмы в Канаде, Великобритании, Германии, Франции, Швейцарии, Австралии и Латинской Америке. Кроме того, Merisel является учредителем дочерней фирмы в Москве – СП "Merisel-Компьютерные технологии". Через свою 55-тысячную дилерскую сеть компания Merisel поставляет 8 000 наименований продукции от более чем 700 наиболее известных мировых производителей аппаратуры и программ. Merisel предлагает все разнообразие платформ IBM-совместимой техники, а также периферию и программное обеспечение к компьютерам Macintosh фирмы Apple, множество типов рабочих станций и многопользовательских систем, работающих в среде UNIX.

Итак, Merisel, не производя ничего, кроме услуг, занимает на мировом рынке средств вычислительной техники одно из центральных мест. Принимая на себя функции сбытовых отделов фирм-производителей и специализируясь именно в этом бизнесе, Merisel обеспечивает более эффективную реализацию продукции. СП "Merisel-Компьютерные технологии" придерживается той же стратегии, учитывая специфику местного рынка, где дилерское звено еще не так развито, как, скажем, в США.

По свидетельству Генерального директора "Merisel – Компьютерные технологии" Михаила Краснова, компания предполагает существенно расширить свою дилерскую сеть в России и других странах СНГ и увеличить число партнеров с 80 до 1 000 в самое ближайшее время. Такие ожидания не кажутся слишком оптимистичными, если принять во внимание тот, например, факт, что ежемесячный объем продаж программ на отечественном рынке через Merisel увеличился с июля по декабрь 1991 года более чем в 30 раз и превысил 1 млн. рублей.

Нетрадиционность посреднических услуг Merisel для отечественного рынка состоит в том, что все необходимое (аппаратуру, программы, любую периферию, включая сетевое обеспечение) дилер может приобрести у одного поставщика ("one stop shopping") и, конечно, в любом количестве. Интерес для дилеров представляет также способ предоставления скидок – не только по количеству, но и по объему закупок, распределенному во времени (концепция "Frequent Buyer").

Эти и многие другие маркетинговые приемы позволяют делать покупки у Merisel дешевле, чем у непосредственного производителя. Этот парадокс (ведь принято считать, что работа с посредником только увеличивает стоимость товара для потребителя) – результат огромных объемов закупок, благодаря которым Merisel получает максимальные скидки у производителя, а также четкой организации доставки, хранения и реализации продукции.

Эффективность закупок у Merisel во многом зависит от уровня технической грамотности и коммерческой компетентности дилеров. Поэтому Merisel не только уделяет много внимания обучению своих дилеров, но и является новатором в этой области. Так, руководитель отдела выставок и рекламы Марина Никитина сообщила о готовящемся специально для дилеров мероприятии под названием "Softeach", которое будет проведено во время выставки КОМТЕК-92. На этом мероприятии, организуемом в форме презентаций фирм-производителей, дилеры получат возможность не только познакомиться с реально предлагаемыми к продаже образцами продукции, но и тут же приобрести их со специальными скидками.

Посетив Merisel в Москве, я получил представление о компании, как об очень серьезном партнере. Остается лишь надеяться, что на нашем региональном рынке появится больше фирм, стремящихся придать ему цивилизованные формы. ■

ИНФОРМАЦИЯ ДЛЯ ПРОФЕССИОНАЛОВ В ОБЛАСТИ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Впервые на восточно-европейском рынке Вы можете оценить возможности одной из крупнейших в мире оптовых фирм в области вычислительной техники

MERISEL SM

Merisel - это поставка по оптовым ценам любого из почти 8000 наименований технических и программных средств более 700 ведущих фирм-производителей из одного источника.

Merisel - это поставка оборудования и программных средств с консигнационного склада в Москве, а также комплекс дополнительных услуг, включающий содействие в оформлении импортного контракта, техническое и информационно-консультативное обслуживание и обучение Ваших специалистов, регистрацию пользователей программных средств и поставку новых версий.

Merisel- это уникальная возможность прямых контактов дилеров с представителями крупнейших фирм-производителей вычислительной техники в Москве на форуме SOFTEACH в период проведения выставки KOMTEK-92 . Кроме того,в программе SOFTEACH распродажа новейших продуктов, лотереи, конкурсы, шоу по-американски.

Контакты:
Merisel: 200 Continental Blvd., PO Box 984,
El Segundo, CA 90245-0984, USA,
(1-213)615-3080, Wendy Zenchyshyn,
International Marketing Division.
Merisel – Компьютерные технологии:
109044, Москва, Крутицкий вал, д.3,корп.2,
телефоны: (095)276-4714, (095)276-4718,
факс: (095)274-0097.

MERISEL

Clipper 5

ВЫБОР ПРОФЕССИОНАЛОВ

Став зарегистрированным пользователем, Вы сможете:

- **обладать полностью русифицированным продуктом и документацией (за рубли);**
- **легально разрабатывать и распространять собственные коммерческие продукты на Clipper;**
- **получать бесперебойные консультации по "горячей" телефонной линии (095) 289-43-00.**

Легальная копия – входной билет в мировое сообщество Clipper

За информацией обращайтесь в СП "Магнит,"
Наш адрес: 127018, Москва, 2-я Ямская, 15.
Телефон: (095) 289-44-77 Факс: (095) 289-34-69

Nantucket®